HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Electrical and Communications Engineering
Degree Programme of Electrical and Communications Engineering

Tuomo Koskenvaara

Decentralization of multimedia content in a heterogeneous environment

Master's thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Technology

Supervisor Professor Jorma Tarhio
Instructor M.Sc. (Tech.) Harri Salminen

Geneva, September 25, 2002

TEKNILLINEN KORKEAKOULU DIPLOMITYÖN TIIVISTELMÄ

Tekijä:	Tuomo Koskenvaara
Työn nimi:	Multimediasisällön hajauttaminen heterogeenisessa ympäristössä
English title:	Decentralization of multimedia content in a heterogeneous environment
Päivämäärä:	25. syyskuuta, 2002 **Sivumäärä:** 68
Osasto:	Sähkö- ja tietoliikennetekniikan osasto
Professuuri:	T-106 Ohjelmistojärjestelmät
Työn valvoja:	Professori Jorma Tarhio
Työn ohjaaja:	DI Harri Salminen

Tässä työssä tarkasteltiin multimediasisällön hajauttamista heterogeenisessa ympäristössä. Toimintaympäristönä toimi Euroopan hiukkasfysiikan tutkimuskeskuksen (CERN) verkko ja Suomen yliopisto- ja tutkijaverkko sekä niiden välinen verkkoympäristö. Euroopan hiukkasfysiikan tutkimuskeskus tuottaa multimediasisältöä, jota voidaan käyttää hyväksi fysiikan opetuksessa. Web University -pilotti on kehittänyt tämän multimediasisällön jakelua vuodesta 1997 lähtien.

Multimediasisällön jakelu vaatii paljon verkkokapasiteettia. Kullakin sisältötyypillä on omat erityistarpeensa jakelujärjestelmälle mutta Internetin kaltaisessa heterogeenisessa ympäristössä näiden tarpeiden täyttäminen voi olla vaikeaa. Tilanteen parantamiseksi on olemassa erilaisia keinoja, joista hajauttaminen on yksi käytetyimmistä. Sen toteuttamiseen käytetään yleensä peilaamista ja erilaisia välimuistiratkaisuja. Viime vuosina kehitetyt sisällönjakeluverkot käyttävät molempia edellä mainittuja menetelmiä täyttämään sisällön siirron erityisvaatimukset.

Työn käytännönosuudessa mitattiin verkon kapasiteettia Euroopan hiukkasfysiikan tutkimuskeskuksessa sijaitsevan multimediapalvelimen ja Suomen yliopisto- ja tutkijaverkkossa olevan asiakkaan välillä sekä suunniteltiin ja rakennettiin hajautusjärjestelmä multimediasisällölle. Mittausten jälkeen todettiin, että Suomen yliopisto- ja tutkijaverkkoon liittyvien käyttäjien kannalta hajauttamiselle ei ole tarvetta. Kapasiteettia on riittävästi, vaikka käyttö kaksinkertaistuisi. Euroopan hiukkasfysiikan tutkimuskeskus reitittää kaiken tutkijaverkkojen ulkopuolisen liikenteen Yhdysvalloissa olevan reitittimen kautta. Tämä saattaa aiheuttaa kapasiteettiongelmia suomalaisille käyttäjille, joilla ei ole mahdollisuutta käyttää Suomen yliopisto- ja tutkijaverkon tarjoamia ulkomaanyhteyksiä. Kyseistä käyttäjäryhmää ajatellen suunnittelin ja rakensin yksinkertaisen, modulaarisen ja siirrettävän hajautusjärjestelmän.

Avainsanat: Hajauttaminen, heterogeeninen, Internet, multimedia

Työn sijaintipaikka:

HELSINKI UNIVERSITY OF TECHNOLOGY ABSTRACT OF MASTER'S THESIS

Author:	Tuomo Koskenvaara
Title of thesis:	Decentralization of multimedia content in a heterogeneous environment
Finnish title:	Multimediasisällön hajauttaminen heterogeenisessa ympäristössä
Date:	September 25, 2002 **Pages:** 68
Department:	Department of Electrical and Communications Engineering
Chair:	T-106 Software Technology
Supervisor:	Professor Jorma Tarhio
Instructor:	M.Sc. (Tech.) Harri Salminen

The aim of this study has been the decentralization of multimedia content in a heterogeneous environment. The environment consisted of the research networks connecting the European Organization for Nuclear Research and the Finnish University and Research Network. The European Organization for Nuclear Research produces multimedia content which can be used as studying material all over the world. The Web University pilot in the European Organization for Nuclear Research has been developing a multimedia content delivery service for years.

Delivering the multimedia content requires plenty of capacity from the network infrastructure. Different content of the material can have different demands for the network. In a heterogeneous environment, like the Internet, fulfilling all the demands can be a problem. Several methods exist to improve the situation. Decentralization of the content is one of the most popular solutions. Mirroring and caching are the main methods for decentralization. Recently developed content delivery networks are using both of these techniques to satisfy the demands of the content.

The practical application consisted of measurements of the network connection between the multimedia server in the European Organization for Nuclear Research and the Finnish University and Research Network, planning and building a decentralization system for the multimedia content. After the measurements, it became clear that there is no need for decentralization of the multimedia content for users that are able to utilise the Finnish University and Research Network. There could be double today's usage, and still there would be no problems with the capacity. However, the European Organization for Nuclear Research routes all traffic that comes from outside research networks through a gateway in the USA. This affects every connection that is made from Finland: users are not able to use the international connection offered by the Finnish University and Research Network. For these users I designed and built a simple, modular and portable decentralization system.

Keywords: decentralization, heterogeneous, Internet, multimedia

Library code:

Preface

Even though it was not an easy solution to come CERN to work on this thesis, it was probably most suitable for me. It was the first time during my studies that I really got some time to study. Working here at CERN has been really rewarding for me and I would like to thank all of the people that have been involved in this work. Special thanks to my supervisor Jorma Tarhio, instructor Harri Salminen, Web University, The Helsinki Institute of Physics and to my family Kati and Merri.

Geneve, September 25, 2002

Tuomo Koskenvaara

Contents

Terms and definitions

ANSI	American National Standards Institute – A voluntary organization that creates standards for the computer industry.
ATM	Asynchronous Transfer Mode – A high-performance, cell-oriented switching and multiplexing technology that utilizes fixed-length packets to carry different types of traffic.
CAD	Computer Aided Design – The use of computers to assist the design process.
CDN	Content Delivery Networks – A set of optimized network elements that serve content.
CEPT	Conference European of Post and Telecommunications – An oraganization that cooperates on commercial, operational, regulatory and technical standardization issues.
CERN	Conseil Européen pour la Recherche Nucléaire – European Organization for Nuclear Research.
CSC	The Finnish IT center for Science.
CSMA/CD	Carrier Sense Multiple Access/Collision Detection – A link layer protocol that is used for network collision detection and resolution. A node detecting other traffic on the network stops transmitting, waits a random amount of time, and then tries again.
DCMI	Dublin Core Metadata Initiative – An organization dedicated to the development of more intelligent resource discovery systems.
DXF	Data Exchange Format – Format used in AutoCAD to store vector files.
EIA	Electrical Industries Alliance – A trade association representing the U.S. high technology community.
FDDI	Fiber Distributed Data Interface – A 100-Mbps token-passing, dual-ring LAN using fiber-optic cable.
FTP	File Transfer Protocol – An application protocol used in file-transfers.
GÉANT	The pan-European Gigabit Research Network.
GIF	Graphics Interchange Format – A widely used proprietary graphic format.

HTML	Hypertext Markup Language – Language that is used to describe hypertext pages.
HTTP	Hypertext Transfer Protocol – An application layer protocol that is used to transfer HTML files and files that are linked to them.
ICMP	Internet Control Message Protocol – A transport layer protocol that provides feedback about problems in the communication environment.
IEEE	Institution of Electrical and Electronics Engineers – An organization composed of engineers, scientists, and students. The IEEE is best known for developing standards for the computer and electronics industry.
IP	Internet Protocol – A network layer protocol that handles the routing of packets.
ISO	International Organization for Standardization – A worldwide federation of national standards bodies from some 140 countries, one from each country.
ISP	Internet Service Provider – A company that provides Internet connections to end users.
ITU-T	International Telecommunications Union Telecommunications Sector – An organization that produces high quality standards covering all fields of telecommunications.
JPEG	Joint Photographic Experts Group – A group of experts nominated by national standards bodies and major companies which aims to produce standards for continuous tone image coding.
LAN	Local Area Network – A computer network that spans a relatively small area.
LZW	Lempel-Ziv-Welch – A good all-purpose data compression technique.
NORDUnet	The Nordic Internet highway to research and education networks.
NAT	Network Address Translation – A method of connecting multiple computers to the Internet (or any other IP network) using one IP address.
NVT	Network Virtual Terminal – A bi-directional character device.
OSI	Open system interconnection – An ISO standard for worldwide communications that defines a networking framework for implementing protocols in seven layers.
PCM	Pulse Code Modulation – A method for digitizing audio.
PDH	Plesiochronous Digital Hierarchy – A technology used in telecommunications networks to transport large quantities of data over digital transport equipment.
RAM	Random Access Memory – A type of computer memory that can be accessed randomly.

RDF	Resource Description Framework – An infrastructure that enables the encoding, exchange and reuse of structured metadata.
RTCP	Real-time Transport Control Protocol – An application layer protocol for controlling the realtime transmissions.
RTP	Real-time Transport Protocol – An application layer protocol that supports realtime synchronization.
RTSP	Real Time Streaming Protocol – An application layer protocol for control over the delivery of data with real-time properties.
SDH	Synchronous Digital Hierarchy – An international standard for synchronous data transmission over fiber optic cables.
SVGA	Super Video Graphics Array – A set of graphics standards designed to offer greater resolution than VGA.
SYLK	Symbolic link – Format used by Microsoft to store vector files.
TCP	Transmission Control Protocol – A statefull transport layer protocol.
TEKES	National Technology Agency in Finland.
TUT	Tampere University of Technology.
UDP	User Datagram Protocol – A stateless transport layer protocol.
URI	Uniform Resource Identifier – The generic set of all names/addresses that are short strings that refer to resources.
URL	Uniform Resource Locator – An informal term (no longer used in technical specifications) associated with popular URI schemes.
URN	Uniform Resource Name – A URI that has an institutional commitment to persistence, availability or a particular scheme specified by RFC2141 and related documents, intended to serve as persistent, location-independent, resource identifiers.
VGA	Video Graphics Array – A graphics display system for PCs developed by IBM.
W3C	World Wide Web Consortium – Consortium that develops interoperable technologies for World Wide Web.

Chapter 1

Introduction

This thesis is based on research perfomed for the Web University pilot project at CERN[1]. In this work I investigate when it is beneficial to decentralize and what part of the content that CERN offers should be decentralized.

The idea for Web University was born in Geneva in 1996. In the beginning of 1997, Web University began as one of the pilot projects of the TEKES-funded Finnish national ETÄKAMU-program. Now the pilot is part of the TUT's[2] coordinated Open Learning Environment-project that is funded by TEKES[3] and companies. Web University offers the possibility for academic society and the general public to attend CERN lectures. Lectures are available on the Internet as digital recordings [1].

Efficient compression algorithms can code multimedia content in a smaller space than before, but at the same time, the amount of content is rapidly growing. With the current growth of computing power and the growing capacity of Internet connections, it is now possible to watch a multimedia presentation almost anywhere in the world. However, sometimes the user or the material is in a place where connections are inferior, or there are so many users that they fill up the connection and the service becomes unavailable.

Decentralization is one way to offer multimedia material to a wider number of users. Decentralization means that at least a part of the multimedia material is copied closer to end user locations. It can be used to secure the usability of the material and load balancing. Decentralization is usually used in software delivery systems like in FTP servers.

Many material distribution systems have a built-in capacity to decentralize

[1]Conseil européen pour la Recherche Nucléaire – European Organization for Nuclear Research.

[2]Tampere University of Technology.

[3]National Technology Agency in Finland.

the material between other nodes running the same software. This kind of system is called homogeneous. The system is heterogeneous when there are different solutions for offering the same service. To be able to communicate in a heterogeneous system, one must be aware of the protocol other systems are using. For that, we need to open specifications and standards. Most distribution systems have their own communication protocol and in most cases that information is not released for public. The Internet itself is a heterogeneous system and there are many protocols available for accessing other computers. HTTP, FTP and RTP are a few examples.

Now the pilot project needs more information. Should it decentralize more material to our Finnish technology partner CSC[4] or not? If it should, what part of the material should be decentralized and when? A third question is how it should technically be done. These are the three main questions that I am trying to answer in this work.

The second chapter deals with different types of multimedia. It provides answers that are requisite to using multimedia. It also explains some basics about multimedia, networking and heterogeneous systems. The third chapter deals with decentralization: what it means, and why and when it is needed. The case study of the Web University is addressed in the fourth chapter. The last chapter includes my conclusions.

[4]The Finnish IT center for Science.

Chapter 2

Using multimedia

In this chapter, I will explain some background information about multimedia. First, I explain the basics of multimedia content. Next, there is a brief explanation of the term "heterogeneous system," and finally a section about network connections.

2.1 Multimedia content

Multimedia – it is a word that has no exact meaning. Everyone seems to know what it means, but no one can explain it completely. Normally, multimedia is understood as a combination of two or more media [2]. This is the definition I will use throughout this paper. Today, the term multimedia is normally associated with computers. Computer multimedia is usually some sort of presentation or interactive program that uses different media to interact with the user. Actually, "multimedia" in terms of the combination of two or more media, has existed for at least 17,000 years, when early man made prehistoric paintings in the Caves of Lascaux. They probably had ritualistic gatherings, where they combined graphics and sounds [3]. The word itself is new, but its meaning has been with us a long time.

Today's multimedia material is normally stored in digital form. Digital formats have the advantage that they do not deteriorate with use like ordinary video or cassette tapes. Multimedia material is used with multimedia-capable computers. A typical multimedia computer supports the playing, editing and recording of multimedia material, as well as communication using audio and video via a network connection [4, p. 23]. With today's typical home computer, a user is able to edit video or audio stream almost in real-time. This is something that a user could not do at home a couple of years ago. This development has enabled growth in production of multimedia material. Now almost every computer user is at least able to use multimedia material with

his or her computer at home or at work. The biggest bottleneck is normally the transfer of the material. According to research done 1998 by Statistic Finland, less than 10% of the population of European countries had a connection to Internet. In the autumn of year 2000 almost 40% of Finnish citizens had an Internet connection from their homes [5]. Most of these connections are still made with dialup-lines that are not suitable for transfer of multimedia material because of their limited bandwidth.

A couple of years ago, most digital material was simply text and pictures. Today, multimedia combines text, graphic, audio and video components. These components are compressed using different technologies. For optimum compression, it is not possible to use the same algorithms for different types of media. Pictures can be compressed with a so-called lossy method. This means that it loses some of its information every time it is compressed. One cannot use this kind of method compressing text. Compression also requires that the end user use the same methods to decompress the material before using it.

2.1.1 Types of multimedia

Digital multimedia material is normally a combination of different types of media content. The media used today are normally text, graphics, video and audio. It is difficult to guess what media will become available in the next ten years. Different types of media are suitable for different purposes. Some are very easy to implement and transfer, while others might require more work.

Pictures were probably the first method of storing information that humankind discovered. Pictures were used to preserve stories in ancient Egypt. Although there is a phrase: "A picture is worth a thousand words", today, pictures are mostly used to support written text.

When an analog picture is converted to a digital image, it is first rasterized in a user-defined number of pixels. Color information is then calculated and rounded to nearest computer-represented value. The number of values used depends on the standard used. This method is called bitmap. Pictures that are made by a computer can have a more mathematical method of storing. This method is called vector.

GIF and JPEG are the most commonly used bitmap presentations. A company called CompuServe defined the GIF format. It works well for pictures with few colors or with sharp edges like cartoons [4, p. 11]. The format was thought to befree for use. However, the compression scheme used, called LZW[1], is claimed to be patented, and its originators are requesting royalties for using it. A new format, called PNG[2], has been developed to replace GIF in the future [6, p.

[1] Lempel-Ziv-Welch.

[2] Portable Network Graphics.

10]. JPEG is a standard defined by ISO[3] that is actually just a compression method that is used for file formats like JFIF. Most users call files compressed with JPEG method "JPEGs" [6, p. 18]. JPEG compression is a lossy method. Most of the popular image compressing schemes, like the one used with GIF, are loss less. Loss less means that no data is discarded during the encoding process. The JPEG compression method has a superior compression ratio compared to most loss less schemes. The scheme is developed to lose only that information which is not easily visible to human eye. It is very suitable for pictures with smooth surfaces and a large color scale. JPEG loses information on every compression.

Vector formats are useful for storing line-based elements and elements that can be composed from them, like lines, arches, polygons and text. Information is stored in mathematical form rather than in pixel values. One advantage of vector format is its scalability. Vector formats rarely compressed at all because of the mathematical presentation. Vector graphics are normally used in CAD[4]. DXF and SYLK are two of the most commonly used vector formats [7, p. 13].

Text has been the major form of communication between computers and humans. Written text is one of the prime forms of asynchronous communication between humans [8, p. 31]. Text has two broad forms: unformatted and formatted. An unformatted text character set is normally limited and the style and size are fixed. A formatted text character set is larger. It has multiple fonts, sizes and styles.

Text can easily be compressed, but requires loss less compression methods. When compared to other media, text uses so little space that there is normally no need for compression.

People have always communicated with voice, so audio is often used in multimedia content. Music and conversations have been with us for ages. Audio was first stored in the memories of storytellers and troubadours. Later, people started writing down those stories with words and notes, in effect converting them to other media. First, sound recordings were analog, being recorded on cylinders made of clay. After that, sound was stored on magnetic tape. By the end of the Second World War, all fundamentals of today's analog audio had been discovered [9, p. 1]. Digitalization of audio made a huge step toward popularity during the introduction of CD-player.

The digitalization process of sound is much like the digitalization of pictures. The sound is sliced into samples of time and rounded to the nearest value. Sound from a CD is composed of 44.100 samples per second, with values of 16 bits, i.e., 65536 different values [9, p. 560]. This method is called PCM.

Computers can also produce sounds artificially. This method can give any possible value to any of the time slices to produce a sound. That is, the time

[3]International Organization for Standardization.

[4]Computer Aided Design.

scale is not fixed. In CD standard, it is fixed to 44.1 kHz, but these are only attributes for the A/D or D/A converter in a computer. The maximum value depends on the hardware and software used. One minute of CD-quality music occupies about 10 Mbytes and takes 24 minutes to transfer through a normal dialup modem line.

Audio can be compressed using various methods. Normally audio is compressed with lossy methods. Modern compression techniques are exploiting the properties of the human ear. They can achieve size reduction by a factor of 12 with unnoticeable loss of quality [10]. There are many compression methods on the market, some are open standards and some are product specific. One of the most common open standards is ISO-MPEG Audio Layer-3, better known as mp3, which was developed in Fraunhofer IIS-A 1987 [11].

Video is used in many modern multimedia presentations. Video is normally a collection of still images that are not independent. When video is displayed on a computer screen, it may occupy the whole screen or just part of it. One still picture of a movie is called a "frame", and the number of frames displayed per second is called the "frame rate". A smooth impression of movement needs at least 16 frames per second. [8, p. 37-38]

A collection of still images requires a great deal of storage space. One minute of SVGA video that has 640 * 480 pixels with a depth of 24 bits requires approximately 50 Mbytes, and this is why compression methods are very important for delivery of video material. There are several methods for compressing video, some of them are standard and some are proprietary. Compression methods are mostly lossy ones, and they try to lose information that is similar from frame to frame. One good example of this is the background of a picture. It can be transmitted once with the first frame, and then only changes in the background are sent. Videoconferencing standard H.261 recommended by ITU-T offers that functionality [4, p. 83].

This method has been further developed for entire pictures. The MPEG standard, developed by the ISO, defines a group of pictures (GOP) structure. Each GOP starts from an intra frame and has N-1 other frames, from which some are prediction frames. The intra frame is compressed using a method similar to that employed in the JPEG. The prediction frames are compressed using the nearest decompressed intra or prediction frame as a reference to estimate the motion. The rest of the frames are decompressed using the nearest decompressed intra and prediction or prediction and prediction frames as a reference to bidirectionally estimate the motion from each frame. The motion in the pictures is stored in a vector variable. Only the vector variables are stored as information for the rest of the frames. An example of a GOP might have one intra frame, four prediction frames and ten motion vectors for calculation of the rest of the frames. [4, p. 87-88]

Computer produced animation can be either a collection of still images or a computer program that is executed during the play. It has different ob-

jects that have preprogrammed parameters prescribing the existence and the location on the screen. In this way, the information about the image can be compressed into a small mathematical representation. If the animation is complex, its computation can consume lots of processing power. [8, p. 38-39]

Metadata can be defined simply as structured data about data. It is descriptive information about an object or resource that can be either physical or electronic [12]. Actually, metadata is not part of the multimedia content, but a necessary part of a functional multimedia environment. From text media, it is quite easy for the user to search for certain phrases. Other media, like audio, picture or video, do not have this advantage. How can the program know what the voice is saying, or even worse, singing? These are the cases for which we use metadata. One form of metadata is the card index catalogue in a library. The information on the card is metadata about a book.

Metadata is normally stored in a database separately from the content itself. Metadata aims to explain the content in a form that can be easily processed by computers. There are many different proposals for standards in this field. One is Dublin Core Metadata IETF RFC2413 produced by DCMI[5] an organization dedicated to the development of more intelligent resource discovery systems. W3C[6] has developed their own recommendation, called RDF.

To create multimedia content, one must combine different media together, and this requires tools. Of course, this can be accomplished with a classical programming language like C++, Java or Visual Basic, but it is more convenient to use languages and tools made specifically for multimedia [4, p. 62-63]. One of the first tools for making multimedia presentations was DOS-based Storyboard Live! [13, p. 34]. Even though it was a DOS-based program, it had a nice graphical interface and supported a mouse. The biggest problem with that system, however, is that you need the whole program in order to use the material made with the program.

Today, there are many different tools for making multimedia content. Normally, they have one program for creating the content and another for viewing it. Viewing the program is usually made free for the public, so that anyone is able to use the content, but must pay for a production program. Viewing programs are usually made as plugins for popular web browsers. Good examples of program pairs such as these are QuickTime from Apple and Director from Macromedia. These both have one program for producers and a plugin for popular web browsers for viewing the content.

There is also a programming language called SMIL[7] that was developed for integrating multimedia material. The syntax of the language is based on XML [14]. Its latest version(2.0) was published in autumn, 2001. SMIL is one of the software-independent solutions for producing multimedia material. See figure

[5]Dublin Core Metadata Initiative.

[6]World Wide Web Consortium.

[7]Synchronized Multimedia Integration Language.

2.1 for an example of an SMIL presentation. CERN produces lectures using SMIL and Sync-O-Matic formats. Sync-O-Matic is a free software that is used for RealMedia™based web lectures [15]. The major difference between these two formats is that Sync-O-Matic is based on HTML and is less configurable than SMIL.

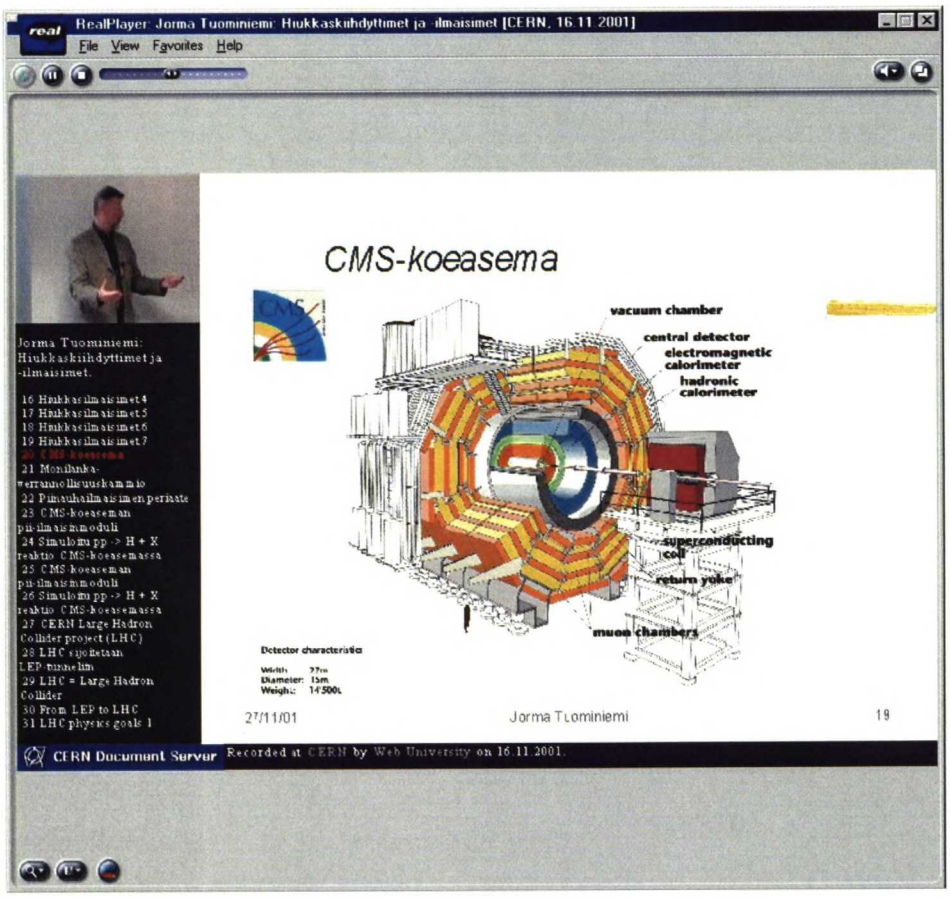

FIGURE 2.1: Picture of a recorded lecture produced with SMIL

2.1.2 Requirements of different mediatypes

Retrieving multimedia content through a network connection is an everyday thing. Requirements for different media types vary greatly. There are time dependent media, like audio and video, and there are time independent media, like text and pictures. In the case of time dependent content, the network must deliver the packages in time and in order. The receiving end-system can ease this problem by using a buffer of a few seconds worth of information at all times. This component will add substantially to the overall latency. Latency is not the biggest problem if it stays under a certain level; problems arise if the latency varies. This variation is called jitter and is described in subsection 2.3.1. For example, if the beginning of a video takes two seconds, there is

no problem. However, if the next picture arrives two seconds after the first is already on the screen, the user can easily notice the latency. Jitter and network throughput are the two main requirements for delivering multimedia content.

Video and audio content can be transmitted live or it can be stored and transmitted later. This has no impact on the requirements of the connection [8, p. 347]; they both need the same quality for proper transport. There are some physical differences in how people hear and see. We are much more sensitive to alterations in audio signal than in visual signals, so the packets delivering audio should have a higher priority [8, p. 369].

Time independent content has smaller requirements for transport connection. However, time independent content may also be used within time dependent multimedia content. Text and image content can be delivered in such a way that it is possible to show part of it, even though the rest of the content has not arrived yet. Text files are normally small and do not need any compression, but picture files tend to be large if they are not compressed.

If the content is compressed, it cannot tolerate as many errors during the transmission as the original because bits before and after the error bit are in many cases dependent on the information of the error bit. In the case of original content, every bit is individual and only that information is lost. [8, p. 369]

Some examples of requirements of the audio and video content in table 2.1.

Content	Original	Compressed	Jitter	Bit error rate
Audio (CD quality)	1.4 Mbps	256 Kbps	100 ms	10^{-4}
Audio (Telephone quality)	64 Kbps	4.8 Kbps	400 ms	10^{-2}
HDTV	2 Gbps	25-34 Mbps	50 ms	10^{-6}
TV (Studio quality)	166 Mbps	3-6 Mbps	100 ms	10^{-5}
H.261 Videoconferencing	–	112 Kbps	400 ms	10^{-4}
Multimedia	–	–	30 ms	–

TABLE 2.1: Specification of the requirements of audio and video content [8, p. 350-364]

2.2 Heterogeneous systems

Heterogeneous systems are those with a range of diverse computing resources that can be near one another or geographically distributed. This is an every day

situation in many computing environments, for example in Internet. There are many different computer hardware solutions running many different operating systems, and they cooperate with each other using TCP/IP protocol stack, which is an open protocol standard and has no license fees [16, p. 19]. Actually, heterogenity can improve efficiency of computation by offering certain solutions for certain problems or tasks [17]. If one solution is designed to perform a certain task, it is in many cases a poor solution to perform different task. This is the advantage of heterogeneity. To be able to cooperate in heterogeneous systems there must be well defined interfaces.

2.2.1 Standards

Long ago, the computer industry had many closed systems where one could only connect devices from the same manufacturer. Today, many standards and open solutions have been developed to correct this situation. Standards are not laws, and no one is obligated to follow them, but they provide an easy solution if everyone is willing to cooperate [18]. TCP/IP -stack and the protocols it offers make up one of the most widespread open solutions and enables different nodes to cooperate through a network. In practice, there is only one other open system network standard: OSI, developed by ISO [16, p. 19]. It is widely used in GSM networks.

There are many international bodies that are concerned with the development of standards. There are many institutes in the field of computer networks, for example ECMA[8], EIA[9], IEEE[10], ISO, ITU-T[11], CEPT[12] and ANSI[13].

All computer programs that are able to communicate with other computer programs made by a different manufacturer are using some predefined protocol for cooperation. One good example of this is web browsers and servers communicating with eachother using HTTP protocol.

2.2.2 Problems

Even though it seems that standards are the key for open solutions; there are some drawbacks. There are many standardization organizations, and they are making different standards for the same purposes and sometimes those standards are incompatible with eachother. This has been the case, for example, in electrical and phone line plugs. Another problem is the time it takes to

[8] European Computer Manufacturers Association.

[9] Electrical Industries Alliance.

[10] Institution of Electrical and Electronics Engineers.

[11] International Telecommunications Union Telecommunications Sector.

[12] Conference European of Post and Telecommunications.

[13] American National Standards Institute.

develop or upgrade a standard. There are many steps involved in developing an official standard, and the process often takes years. Many times manufacturers and researchers develop their own de-facto standard before the official standard comes out, and if the official one is not compatible with the manufacturer's, it usually remains unsupported. A standard may also swell because every organization involved its development wants to add something of their own. It may then become very difficult to implement all the necessary features to fulfill the requirements of the standard.

2.3 Network connection

Multimedia material is delivered in digital form. It can be delivered through the Internet or any other network. In some cases it can be delivered through removable media like CD-ROM or removable hard disk. In this section, I will focus on network delivery.

There are currently two main ways for connecting to a network. A computer can be connected to a network permanently or on demand. For on demand connections, known as "dialup connections", the user normally pays for the time he/she stays online. Initial costs of dialup connections are lower than for permanent connections because most people already have a phone line. On demand connections are usually slower than most permanent ones, and the user must always open the connection before using it. Main network equipment for using dialup connections is analog or an ISDN modem.

A permanent connection offers the user access to the network at all times, and usually a greater bandwidth. Connections are offered with a flat rate, but a certain amount of usage is necessary before it becomes advantageous in comparisonwith on demand connections. Permanent connections also bring new security threats. Because the computer always connected to network, it is always vulnerable to attacks. ADSL, cable modem and LAN techniques are used for making a permanent connection to the end user's location.

2.3.1 Connection characteristics

The way in which to use multimedia content through network connections depends on the characteristics of the connections. The main characteristics of network connections are [8, p. 321-322]:

- The throughput
- The transit delay
- The jitter

- The error rate

Throughput is a parameter that describes the rate at which two end-systems can exchange binary information. This definition has two parts hidden within it. The first is the network throughput that describes the bit rate that the network is able to transfer. The other is the end-systems throughput, which describes the bit rate that the end-system is able to process. The unit used to express the throughput is usually a number of bits per second (bps). More practical units are the kilobit, the megabit and the gigabit per second. Access speed of a Basic Rate ISDN connection is 64 Kbps at best, and for an Ethernet LAN, it is 10 Mbps, 100 Mbps, 1 Gbps or 10Gbps depending the Ethernet technique used. Throughput is sometimes referred to as data rate or transfer rate. It is sometimes also called bandwidth, which is incorrect, because bandwidth is a physical property of certain transmission media. Bandwidth refers to a range of frequencies that can be transferred through that medium. Information is transmitted through media with pulses. The rate of the pulses is dependent on bandwidth. The pulse rate is different from the bit rate. One pulse can hold zero or many bits, depending on the coding method. One physical medium can transfer one or more connections at the same time, depending on the transmission technique used. If there is only one connection on that medium, bits can be transmitted at the access speed. If the medium supports more than one connection at a time, the throughput of the medium is shared between the different connections. Conventional Ethernet LAN is a good example of a medium that has an access speed of 10 Mbps, but that supports many connections. Thus, if there is more than one connection at a time, the access speed differs from the bit rate. The end-system can produce constant traffic at the same bit rate or the rate can vary. There are few applications that produce a constant bit rate (CBR). Variable bit rates (VBR) produce bursts for the network. This is characteristic of data streams. VBR traffic has four interesting properties: the peak bit rate (PBR), the mean bit rate (MBR), the peak duration and the ratio between the MBR and PBR. [8, p. 322-326]

Transit delay is the time that elapses between when the transmitting end-system sends the block of data and when the receiving end-system receives it [19, p. 374]. This is also called latency. Latency can vary between networks and even between directions; some have shorter latency than others, but no network can transmit instantaneously. A physical property of signal propagation on electrical or optical medium requires that it take time to transfer signal through the medium. Transmit delay can be divided in to three parts [8, p. 327]. Access delay describes the time that the source must wait for the medium to be available. Transmission delay is the time it takes to transmit the block of bits that depends only on access speed. Network transit delay is the time that elapses when transferring one bit from the transmitting end-system through the medium to the receiving end-system. Usually, the more interesting metric is the round trip delay, which is the time elapsed between sending a first bit of a data block and receiving it by the same end-system after the block has been echoed by the destination end-system.

The delay can vary from time to time, and this variation is called jitter [20]. It is the most critical characteristic of a network supporting streaming multimedia. Jitter depends on the physical properties of the network medium, structure of the transmission network, current load of the network, temperature, etc. This means that the attribute has a statistical nature; it cannot be calculated or predicted precisely. If the bit rate over an end-to-end network connection has a guaranteed bit rate, and if the value of the delay variance is small and guaranteed, the system is called isochronous.

Transferred data may change during the transfer. Some bits may change their state or even get lost. State changing is a very rare error, but losing data is more frequent and can happen during congestion. Data can also be duplicated, but this is also very rare. The most frequent problem is out-of-order delivery. There can be alternate routes between two end-systems, as is the case with Internet. The routes can change depending on the congestion or failures in the network, so the data that was sent first can arrive at the end-system after the data that was sent later. There are three main metrics for error rates: the bit error rate, the packet error rate and the packet loss rate. The bit error rate is a frequent metric to express the quality of transmission media. It is the frequency of residual erroneous bits. The packet error rate indicates the rate of lost, duplicated or out-of-order packets. The packet loss rate counts packets lost in time. These errors can be detected, noticed and sometimes recovered. Some systems simply detect the error and discard the erroneous packet. Some others, like X.25, may notify that the packet has been lost. Normally the sending end-system decides what to do in the case of an error. Errors can be detected by adding some precalculated check bits in the transferred code. The receiver calculates the same code and checks that the check bits are correct. If incorrect, it can reject the packet. It can then notify the sender of the rejection. If not, the sending end-system must be aware of the sent and acknowledged packets. If the sender does not get the acknowledgement message from the receiver by a certain time, it assumes that the data never got there and tries to send it again. This method is called retransmission. Another technique is similar to the check bit method. For this technique, there are more check bits that are calculated in a certain way so that a computer can produce the correct data, even if some of the bits have changed during the transmission. [8, p. 332-334]

In the case of multimedia content delivery, the ability to multicast in the network is a huge advantage. Multicasting means that the data can be sent only once, even though there are many recipients. It is duplicated only when the two or more end-systems have different routes. [21]

Two other network performance parameters are connection setup delay and priority schemes. The connection setup time depends on the connection technique used. The Internet does not support any priority or quality of service (QoS) features at the moment. Some parts of the Internet might have some priority or QoS features, but they do not work globally.

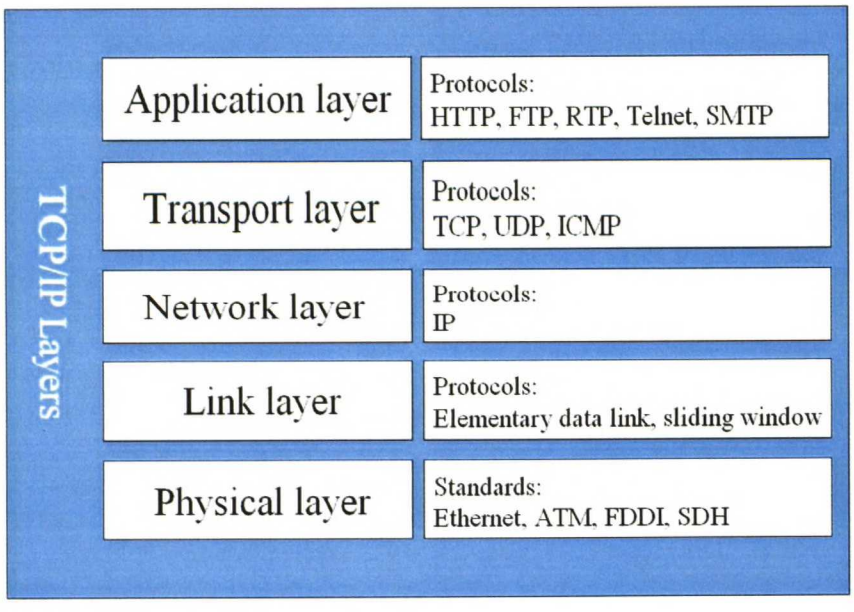

FIGURE 2.2: Layers of the TCP/IP stack

2.3.2 TCP/IP networks

The most commonly used transfer protocol is IP[14], which is the base protocol of the Internet. It is designed for use in interconnected systems of packet-switched computer communication networks [22]. It transmits blocks of data, called datagrams, from sources to destinations, which are identified by fixed length addresses. The length of the address depends on the version of IP used.

Actually IP is just a protocol used in the TCP/IP protocol stack. TCP/IP comes from two acronyms, TCP, for Transmission Control Protocol, and IP, for Internet Protocol. TCP handles packet flow and error recovery between systems and IP handles the routing of packets. All modern networks are now designed using a layered approach; TCP/IP protocol stack is shown in figure 2.2. Each layer represents a predefined interface with the layer above it. By doing so, a modular design can be developed so as to minimize problems in the development of new applications or in adding new interfaces. Each layer offers services to the layer above, hiding the details of the actual implementation behind the predefined interfaces. Every layer communicates with the same layer on the other end of the communication channel. The conventions used in communication are known as protocols [19, p. 10]. Each layer has its own protocols. TCP/IP has five layers, and each of them has many protocols or standards, which are described in the following subsections.

Many different application layer protocols have been developed for different purposes. The application layer provides services for application programs. It

[14]Internet protocol.

offers tools for authentication, compression and end user services. When the connection is established, the application layer protocol is used to determine, for example, error recovery procedures, data integrity and privacy procedures. Below are explanations of some of the most common application layer protocols and a few others that are related to this work. They are also described briefly in table 2.2.

- Virtual terminal protocol (telnet) was developed for a fairly general bi-directional, eight-bit byte oriented communication facility. The primary goal is to allow a standard method of interfacing terminal devices and terminal oriented processes. The three main ideas of telnet are: the concept of a network virtual terminal (NVT), the principle of negotiated options and a symmetric view of terminals and process. The network virtual terminal is a virtual device that provides a standard network wide; an intermediate representation of a canonical terminal. All local device characteristics and conventions are mapped to fulfill the NVT specifications. All traffic between hosts is transferred in NVT form. Negotiation takes place in the beginning of the connection. If the other host is willing to provide additional services over and above those available in NVT, it may ask from another end possibility to use them with a special command. The receiving end replies with an acknowledgment. This way, there are many sophisticated applications that have elegant services. The prevention of loops made by the symmetric negotiation process is simple. It has three rules: parties may only request a change in option status, if the receiving party requested option is already active it sends no acknowledgment, and after the negotiation the other party must insert a command in the data stream at the point where the option is desired to take effect. Telnet uses Transmission Control Protocol (TCP) to transmit data interspersed with telnet control information. [23]

- Secure SHell (SSH) provides more functionality than telnet and it provides strong authentication and secure communications over unsecured channels. One may also tunnel other protocols through it and that way guarantee security of the connection even if the tunnelled protocol is unsecured.

- File Transfer Protocol (FTP) was developed for file sharing, indirect or implicit use of remote computers, to shield the user from variations in file storage systems among hosts and to transfer data reliably and efficiently. It uses two connection schemes, one for control and other for data transfer. The control connection uses the telnet protocol. The control connection is used to set up the parameters and to issue a file system operation. The actual data is transferred through another connection that is opened only for that transmission. [24]

- Rsync remote-update protocol is a mirroring protocol that allows for more efficient transfer of modified files than FTP does. The protocol

allows rsync-application to transfer just the differences between two files across the network, using an efficient checksum-search algorithm [25].

- Simple Mail Transfer Protocol (SMTP) was developed to transfer mail reliably and efficiently. It is independent of the lower layers and only requires a reliable ordered data stream channel. A special feature of SMTP is its capability to relay mail across a transport service environment. [26]

- Hypertext Transfer Protocol (HTTP) is a generic, stateless protocol that is used for many tasks beyond its use for hypertext, through extension of its request methods, headers and error codes. The World Wide Web uses HTTP. The first version, referred to as HTTP/0.9, was a simple protocol, capable of raw data transfer across the Internet. The HTTP/1.0 version offered the possibility of meta information through MIME-like messages. However there were still problems with hierarchical proxies, caching, the need for persistent connections and virtual hosts. Even though some applications that call themselves HTTP/1.0 compliant, they might have implemented those features incompletely. HTTP/1.1 is a current version of this protocol; it uses a Uniform Resource Identifier (URI), that can be Uniform Resource Locator (URL) or Uniform Resource Name (URN), to identify the resource. The protocol message format is similar to that which is used with SMTP. HTTP is used also as a generic protocol for communication between other Internet systems. [27]

  HTTPS was developed for secure end-to-end HTTP connections. It uses three subprotocols to set up the connection. Using these subprotocols, it determines the cryptosystem, keys, error reporting and authentication used. A variety of cryptosystems are supported. [28]

- Real-time Transport Protocol (RTP) provides transport functions that are suitable for applications transmitting real-time data, e.g., video and audio. Services include payload type identification, sequence numbering, time stamping and delivery monitoring. RTP can be used in a multicast or unicast environment. It does not include resource reservations or quality-of-service guarantees, but relies on lower-layer services to do so. Controlling and identification of the data transport is done with Real-time Transport Control Protocol (RTCP) to provide scalability to large multicast networks. RTP and RTCP are designed to be independent of layers used below the application layer. [29]

- Real Time Streaming Protocol (RTSP) is a protocol for control over the delivery of data with real-time properties. It provides an extensible framework to enable controlled, on-demand delivery of real-time data. The data source can include both live data feeds and stored clips. It is intended to control either a single or several time-synchronized streams of continuous media. RTSP typically uses other protocols to deliver the continuous stream, although this is also possible by interleaving the continuous media stream with the control stream. RTSP acts as a network

remote control for multimedia servers. RTSP has its own session mechanism, which is maintained by the server. Both the RTSP client and server can issue requests to the other end system. Syntax of the RTSP is similar to HTTP/1.1 and the extension mechanism of HTTP can in most cases also be added to RTSP. [30]

Protocol	Usage	Security	Encryption	Port
FTP	File transfers	login & password	no	20 & 21
telnet	Virtual terminal	login & password	no	23
SSH	Secure connections	login & password	yes	22
SMTP	Mail delivery	login & password	no	25
HTTP	Web surfing	–	no	80
HTTPS	Web surfing	–	yes	443
RTP	Real-time content delivery	–	no	–
RTSP	Real-time content control	–	no	554
rsync	remote-update protocol	login & password	no	873

TABLE 2.2: Specification of the application layer protocols

The transport layer provides the interface between the communication network and the application layer. It is the layer that gives the user the option to choose a certain level of quality from the network itself, and it is designed to keep the user isolated from the functional and physical aspects of the packet switched network [31, p. 75]. It wraps data from the application level into segments (packets) that can be transported through the packet network. Each endpoint is defined by a port number, and in the case of connection-mode protocol, it is implemented as a finite state machine. Multiplexing capability is offered through port numbers. Each connection is defined with a four tuple (sending host, sending port, receiving host and receiving port). Some protocol implementations may also offer multicast potential on this layer.

- Transmission Control Protocol (TCP) provides a reliable communications stream on top of the unreliable packet Internet Protocol (IP) and is typically used by applications that require guaranteed delivery [32]. It is a sliding window protocol that provides handling for both timeouts and retransmissions. TCP uses sequence numbers with every packet in the data stream to ensure delivery in right order. The connection is started up and closed down with an exchange of special segments between two hosts.

  TCP segments consist of header, options and payload components. The header is always 20 bytes, while the option and the payload length may vary. The header includes ports used on both ends, sequence number,

acknowledgment number, length of the header, flag bits, window size, checksum and urgent pointer. Sequence and acknowledgement numbers are used for ordering the stream; the header field includes the length of option fields if they are used. Flag bits are used for special purposes, e.g., to indicate if the urgent data pointer is valid. Window size indicates the space that the receiver has available for the storage of unacknowledged data and the Checksum is used to validate transmitted data. It is calculated over the header and the payload.

- User Datagram Protocol (UDP) provides non-guaranteed datagram delivery with minimum overhead. It actually gives applications direct access to the datagram service of the IP layer. UDP is used by applications that do not require the service level offered by TCP, or that need to use communication services that are not available with TCP, like multicast or broadcast delivery.

  The only services UDP provides over IP are check summing of data and multiplexing by port number. Thus, the application program using UDP services must deal directly with end-to-end communication problems that a connection-oriented protocol would have handled. Applications that require reliable ordered delivery of data stream should use TCP. [33]

- Internet Control Message Protocol (ICMP) provides an easy way to report errors and failures. Since it delivers only error and diagnostic data, it is a component of every TCP/IP implementation, and the structure of the ICMP protocol header is fixed to improve the efficiency. The header is composed of four fields: type, code, checksum and miscellaneous field. The type field indicates the function of the message used, for example echo reply in the case of ping, or destination unreachable in the case of a lost network node. The code field can contain code for subfunctions within a type. The checksum is computed over complete ICMP message, and the miscellaneous field can contain 32-bit information for miscellaneous purposes. [34]

The network layer defines an interface for the user of an end system to connect into a packet switched network. The connections on this level are made from node to node and the routing between nodes is performed on this level also. It is something like the postal system. It allows you to address a package and drop it in the system, but there is no direct link between you and the recipient. It is quite detailed and rich layer. In TCP/IP world the only protocol on this layer is Internet Protocol (IP). Connectionless IP establishes a virtual connection between two hosts so that they can send messages back and forth for a period of time. It fragments packets from a layer above if necessary.

Currently, the most widely used version is IPv4, but version 6 is just around the corner. IP specifies the format of packets and the addressing scheme. The current version uses 32 bits for an address. The addresses have been divided to four family groups. This is defined more clearly in picture 2.3. Class A

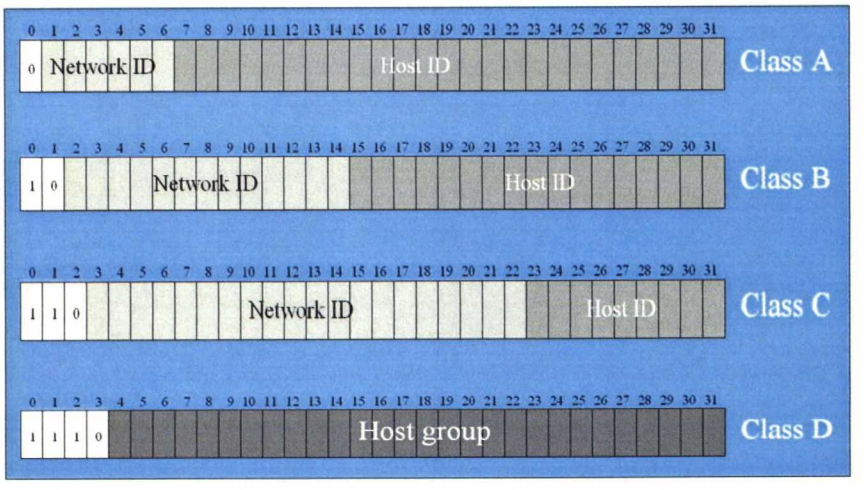

FIGURE 2.3: Internet address types

is for the largest companies and organizations, class B is for medium size organizations and companies. Class C is for small companies, and class D is used as multicasting addresses. [34, p. 23]

This division previously caused loss of network addresses, especially of class A and B, but today the main problem is the length of the address field. A number of technologies have been developed to overcome this problem. These technologies, such as Network Address Translation (NAT), have postponed IPv6. IPv6 has a 128-bit address space that has been divided differently. The main part of the header is simpler than in version 4, and it can be extended with optional parts. This makes it faster to route because of the fixed and simple main header fields.

The IP packet consists of a header part and a payload part. In version 4, the header contains the version number, length of the header, type of service and length of the packet, including the header, identification field, flag bits, fragment offset, time to live, transport protocol, header checksum, addresses of the receiver and sender, possible options and padding. The version number is 4 in this version. Length field specifies the length of the IP protocol header. There is an option for describing the service level in IP header, but it is rarely used because most of the implementations do not process this field. The packet length field contains the length of the whole packet including the header. Since it is only a 16 bit field, the maximum length of an IP packet is 65535 bytes. The identification field is a unique number, like a counter, which is created by the sending host. It is used for reassembly of fragmented pieces. There are only two flags: one denies fragmentation and the other tells that the packet is a piece of fragmented packet. The fragment offset field is used the in case of fragmentation; it contains the position of the submessage contained in the packet relative to the beginning of the whole message. In the time to live field, the sending host can specify how many seconds the packet stays alive in the

network. Every node in the network decreases the value of this field at least 1 second; even if the processing time is less than 1 second. If the value of the field is then zero, the packet must be discarded. The transport protocol field contains information about the payload. For example, for UDP the number is 17, and for TCP it is 6. The header checksum is counted over header, so no checksum is counted over payload. Source and destination addresses were described above and the option and padding fields are used for special purposes. Padding is used to fill the length of the header so that the number of 16 bit words in the header is always a multiple of 4. [34, p. 19-23]

The task of the link layer is to ensure the reliable transmission of information packets and to address stations connected to the transmission medium [34, p. 7]. The link layer accomplishes this task by having the sender break the input data up into data frames (typically a few hundred bytes), transmit the frames sequentially and process the acknowledgment frames sent back by the receiver. Since the physical layer merely accepts and transmits a stream of bits without any regard to meaning of structure, it is up to the data link layer to create and recognize frame boundaries. This can be accomplished by attaching special bit patterns to the beginning and end of the frame. If there is a chance that these bit patterns might occur in the data, special care must be taken to avoid confusion.

Another issue that arises in the data link layer is how to keep a fast transmitter from drowning a slow receiver in data. Some traffic regulation mechanism must be employed in order to let the transmitter know how much buffer space the receiver has at the moment. Frequently, flow regulation and error handling are integrated. Carrier Sense Multiple Access/Collision Detection (CSMA/CD) is the protocol used in the case of Ethernet. The sender listens to the medium, and when there is no traffic it sends the packets to the medium. At the same time, the sender continues to listen the medium, and if there is somebody else sending at the same time, it starts to send a random bit pattern for a short period. Then the sender waits a small random time interval before it tries to send the original packet again. [16, p. 280-281]

If the line is used to transmit data in both directions, this introduces a new complication that the data link layer software must deal with. The problem is that the acknowledgment frames for A to B traffic compete for the use of the line with data frames for the B to A traffic.

- Ethernet, A local-area network (LAN) architecture was developed by the Xerox Corporation in cooperation with DEC and Intel [4, p. 132]. It can use a bus or star topology and supports data transfer rates of 10, 100, 1000 and 10.000 Mbps. The earliest specification served as the basis for the IEEE 802.3 standard, which specifies the physical and lower software layers, described later in this chapter. Ethernet uses the CSMA/CD access method to handle simultaneous demands. It is one of the most widely implemented and used LAN standards. Faster versions are compatible with slower ones and they are developed using the techniques

that worked well with slower versions [35, p. xiii].

- Token Ring, developed by IBM, uses a logical ring structure to connect the computers. The data rates for a Token Ring are 4 Mbps or 16 Mbps and 100 Mbps for a fast Token Ring. One ring can connect up to 250 computers. It used to be more efficient than Ethernet on lower speeds, but the market share stayed lower and it has actually disappeared from the market. [4, p. 136]

  It used a special technique called Token passing for accessing the medium. A token, which is a special bit pattern, travels around the circle. To send a message, a computer catches the token, attaches a message to it, and then lets it continue to travel around the network.

  With the IEEE 802.5 standard, the IEEE standardized the IBM Token-Ring specification.

- Fiber Distributed Data Interface (FDDI) was originally developed for connecting mainframe computers and high-speed peripherals. Development of the standard started in 1982 and took about 10 years to complete[4, p. 140]. It uses optical fiber as a medium and operates with a speed of 100 Mbps. FDDI networks are token-passing networks. It was primarly used as a backbone for campus networks, but today it has been replaced with Ethernet.

- High-Speed Serial Interface (HSSI) is a serial interface that supports transmission rates up to 52 mbps. HSSI is used to connect routers on local area networks with wide area networks. It can also be used to provide high-speed connectivity between LANs, such as token ring and Ethernet. It was developed by Cisco Systems and T3plus Networking.

- Asynchronous Transfer Mode (ATM) was developed to carry any kind of payload. It uses very short, fixed size packets that can be switched by changing the hardware. ATM operates with speeds up to 2,5 Gbps using optical fibre and up to 25 Mbps using twisted pair cable. [4, p. 161]

  Whenever the data transfer begins, ATM determines if the necessary requirements are available and then creates a fixed channel between two points. This makes it easier to track and bill data usage, but it makes it less adaptable to quick changes in network traffic.

  There are four different types of ATM service: Constant Bit Rate (CBR), which is analogous to leased line, Variable Bit Rate, which provides specified throughput capacity but does not send data evenly, Unspecified Bit Rate (UBR), which does not guarantee any throughput levels and Available Bit Rate (ABR), which provides a guaranteed minimum capacity.

- Frame relay is a lightweight packet switching protocol, which was submitted to the ITU in 1984. It uses virtual connections that can be multiplexed over the same access link [8, p. 471-472].

In the TCP/IP communications model, the physical layer defines the mechanical and electrical interface to the physical medium [16, p. 15]. For example, this layer determines how to put a stream of bits from the upper (data link) layer onto the pins for an optical fiber transmitter, a radio carrier or a parallel printer interface. It is usually a combination of software and hardware and may include electromechanical devices but it does not include the physical media as such.

- RS-232C is a standard interface approved by the EIA for connecting serial devices. Most personal computers have an EIA-232 port for connecting a modem or other device and it is still the most common standard for serial communication.
- IEEE 802.3 is a standard specification for Ethernet, a method of physical communication in a LAN that is maintained by the IEEE. 802.3 specifies the physical media and the working characteristics of Ethernet. The original Ethernet supports a data rate of 10 Mbps and specifies these physical media:
  - 10BASE-2, Thinwire coaxial cable with a maximum segment length of 185 meters
  - 10BASE-5, Thickwire coaxial cable with a maximum segment length of 500 meters
  - 10BASE-F, optical fiber cable
  - 10BASE-T, ordinary telephone twisted pair wire
  - 10BASE-36, broadband multi-channel coaxial cable with a maximum segment length of 3,600 meters
- Plesiochronous Digital Hierarchy (PDH) is a nearly synchronous data transmission over digital transport equipment. It is now being replaced by SDH equipment. It allows transmission of data streams that are nominally running at the same rate, but allows some variation on the speed around a nominal rate. It uses transmission rates of up to 565 Mbps [16, p. 74].
- Synchronous Digital Hierarchy (SDH) is an international standard for synchronous data transmission over optical fibres. It is compatible with synchronous optical network (SONET), which is used in North America. All connected equipment is synchronized to a master clock, and the basic transmission rate is 155.52 Mbps, which is known as STM-1. There are also higher rates, such as STM-4 (622 Mbps) and STM-16 (2,4 Gbps). [16, p. 76]

Chapter 3

Decentralization

Decentralization means that part of the content is either placed or copied to different physical locations. There are three main methods for decentralization. The first method was one of the primary goals of ARPANET, an ancestor of the Internet: that the content be decentralized around the net to different nodes. No node has all of the important content. In case of malfunction in some node, all the rest of the important content is still available and usable. Because Internet protocols enable this on a technical level, there is no need to research this any further in scope of this work. The second method is caching, which is used by ISPs and end users to reduce network latencies and network capacity demands. The third method is mirroring the content. It is a bit like caching, but is done before the end users request the content. The implementation of the second method is quite straightforward. It works like a cache system in memory so there can be many levels of caches, but they do not need to know each other. There are certain differences in memory cache and webcache; these will be described in section 3.2 [36]. Mirroring is a more complicated method. Of course, it would be easy just mirror everything to everywhere, but that would consume too much disk and network capacity. What, then, should be mirrored? In caching, the end user makes this decicion, but in mirroring the decicion has to be made in advance. The mirroring operation always consumes network capacity, so it must be done when there is the least network usage. All of these questions are dealt with in the following sections.

3.1 Why decentralization is used

Decentralization was first used to preserve usability in the case of a failure in some node. Today, decentralization is mainly used for load balancing, for usability, saving network capacity and decreasing the latency. Load balancing, shown in picture 3.1, is one of the top reasons because of the incredible growth

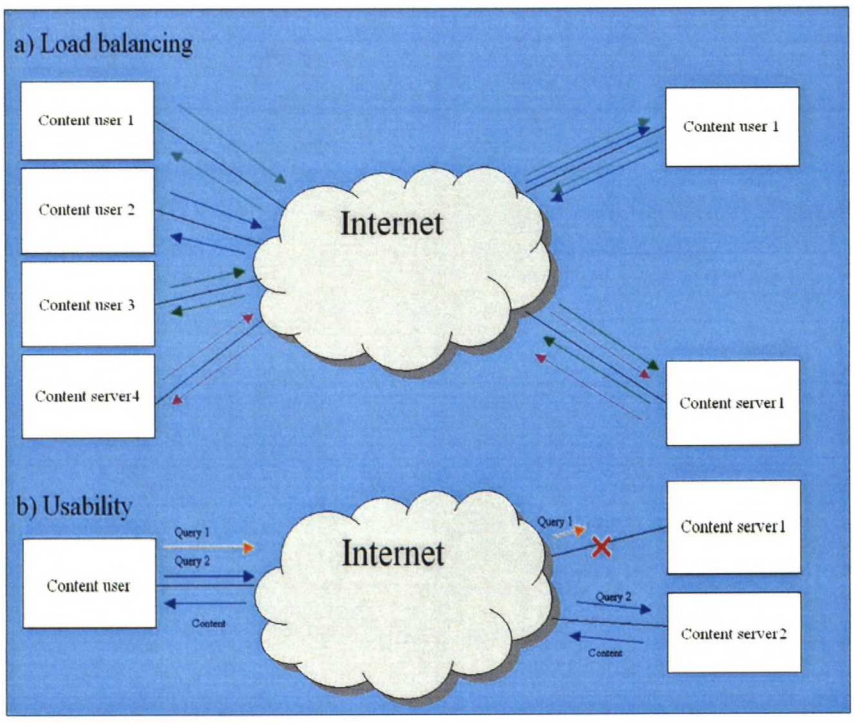

FIGURE 3.1: Ensuring usability and load balancing

in network usage. It is cheaper to improve capacity by adding more nodes to network than it is to increase the capacity of one node. This kind of solution is more scalable for content delivery [37].

With decentralization, an administrator can ensure usability by offering the same content on different nodes. If one node is malfunctioning, users can be advised to use another node, as shown in figure 3.1. As figure 3.1 shows, it is normally more efficient to contact the nearest content server. This also saves international network capacity, and latencies should vary less if the content is available in the user's home country. International network capacity is usually the bottleneck of the capacity for many ISPs.

ISPs do their own decentralization by caching content that users are requesting from outside the ISP's network. The next time the user wants to use the same content, it can be delivered to the user directly from the ISP's own cache memory, so there is no need to use another network.

3.2 Methods for decentralizion

Mirroring, caching and content delivery networks are the main techniques for replicating content. All of these methods are mostly being deployed by content

providers and ISPs. The end user can also use caching. This is normally used in modern network programs, such as web browsers. They cache all the information they retrieve so the next time the user wants to use the content, the browser does not have to retrieve it through the network, but simply checks the validity of the content.

3.2.1 Caching

Caching is a technique for reducing network latencies and bandwidth demands. Caches are deployed at network boundaries, typically at entry points to a network [38]. When a user requests some content from the network, a request is sent to the cache server to verify that it can be found there. If not, the cache server can request the content from the address given by the user and deliver it to the user. The cache server stores the new content locally so that it will be available for future access [36].

The cache holds the retrieved content for a certain period of time. The validity of the object might be checked on every delivery. The cache sends a request for validation of the content from the original content server. If the local content is still valid, it can be delivered to the client immediately. The content might also hold and extra marker indicating that it cannot be cached or that the cached object is valid only certain period. In this case, the cache does not need to check the validity from the original content server. If the object is not cacheable, it will not be stored in the cache at all and will be retrieved from the original content server on every query [39, p. 68]. Content that has a defined validity period can be delivered with out checking the validity from the original content server during the period. There are also some special methods for more detailed cache administration.

Every cache has a replacement policy that normally uses some algorithm. The algorithm might use different metrics for evaluating which element has to be replaced.

There are many levels of caches. For example, every web browser has an internal cache for storing pictures and other downloaded content. Organizations can have their own cache servers for reducing external network usage. This is probably the same reason why every ISP has its own cache server. The cache can be a hardware or a software implementation [40]. Both versions have several manufacturers. They use open protocols for state queries with the original content servers. The content can still be stored in a cache even if it has been removed from the original content server.

The cache can be transparent or nontransparent to the user. Nontransparent caches require different methods for making the client software aware of the existence of the cache. Explicit client configuration and browser autoconfiguration are the most common solutions for this. For explicit client configuration,

every client is explicitly configured to send requests to a cache instead of to the origin servers. For browser autoconfiguration, the browser is configured to download a certain URL every time it is started. The URL holds an autoconfiguration file that holds information about the cache and other things. [41]

Transparent proxies monitor traffic in the network and pick up requests for certain ports (Application layer switch) or addresses (Network layer switch). Because all the traffic then goes through these switches, each request can be processed and the necessary content retrieved if it is not yet stored locally. A network layer switch that has cache capabilities is much faster than an application layer switch because the application layer switch must process the IP packetsin addition to the header. [39, p. 99-117]

3.2.2 Mirroring

Mirroring means placing duplicates of the original content on other machines scattered around the network. A content provider can organize the duplication of the content, or the provider can instruct others to make their own duplicates if needed. Sometimes the content provider wants to prevent others from mirroring their content, and this can be achieved quite easily using the same principles as with the content that is not cacheable. When the content provider organizes the duplication, it can easily offer locations of all the duplicates to the users. If others are doing the duplication of the content, it is normally hard to keep an account of the duplicates. There can also be a problem with the validity of the duplicates. The question is how to inform all administrators of the duplicates about the changes. The content provider should allow others to duplicate only the permanent content, like articles or recordings.

If someone knows of some or all of the duplicates, one can offer the selection of a duplicate that can be done nontransparently or transparently. In the nontransparent solution, all the mirrors are shown to the client, who can choose the best one. However, the client does not know the load situation in the system. Normally these systems are providing geographical information about where the mirror is located. Many transparent solutions have been developed during past years. They can be divided into two classes; Content-blind and Content-aware mechanisms [39, p. 245].

The content-blind approach requires that every mirror is able to process every request for the web site. There are several methods for redirecting a client to a content server. One of the oldest methods is to use DNS server to redirect the client's request [40]. When the client requests an address, e.g., www.company.fi, the request will be redirected to www1.company.fi. The next request will be redirected to www2.company.fi and so on. The company can have N number of web servers. This method is not very effective. It cannot do real load balanding, it simply rotates the next query to the next server in

the chain. It also has no method for recognizing server failures.

A more advanced solution is to use a balancing switch that works on the IP level. It is like a virtual gateway to the web farm: all traffic goes through the switch. It uses a special database to keep track of the ongoing sessions. It is also better able to monitor the state of the machines. [39, p. 233-234]

The content-aware solution is working on an application level, so the query is parsed and the redirection decision is made from the query not just based on the IP level header [42]. This can be implemented on software or hardware. The software solution can be done by the client or server. The hardware solution is normally included in the switch.

3.2.3 Content delivery networks

Content delivery networking is a new way for content providers to improve the quality of service provided to end users. It is easy to mistake CDN for a coordinated caching system [43, p. 146], but it is actually a network that is optimized to deliver specific content [44]. The structure of the network can differ depending on the content that is delivered through the network.

CDN promises two benefits to content providers; global reach and flash event protection. For CDN, the servers are spread all over the globe. Using these special networks, the content provider can deliver the content easily to nearby possible users. Flash event protection is a cure for sudden increases in demand. In this case, the CDN can redirect part of the request to a spare server. [39, p. 247-248]

The content delivery network can technically be implemented in two ways: via the overlay approach and the network approach. In the overlay approach, special servers or caches in the network handle the delivery of certain content, and the core network infrastructure itself has no active role in delivery. Switches and routers are used to build the delivery network in the network approach based on predefined policies. [44]

CDN can be owned by multiple ISPs or it can be owned by one company that does not offer its services externally [39, p. 249].

3.2.4 Tools for decentralization

Previously, most mirroring was done using normal operating system tools such as FTP, rcp, etc. The main advantage to these tools was that they were available with the operating system and were known to be very reliable [45]. Today, several tools have been developed for content decentralization. Some of them are open software, while others are some proprietary. Open software

normally uses an open protocol for transferring the information needed.

As previously mentioned, caching can be used in many levels. Some cache implementations can be found from web servers, e.g., from apache and CERN httpd. Hardware solutions are also popular. A switch or a router can also have a cache implementation. All traffic is usually routed through a certain switch or specific packets can easily be rerouted through a caching switch. There are also special software solutions that were developed for caching purposes. One popular caching server is Squid. Software solutions are normally more flexible than hardware solutions. Flexibility usually comes with lower efficiency and greater maintenance needs. Many clients that use the network, like web browsers and music players, can have a built-in cache to reduce network traffic.

Mirroring is often much simpler than caching. One does not need to worry about validity periods. When mirrors are established, the updating scheme is decided and the system follows it. The question is more about the bandwidth consumption. There are protocols that are developed for mirroring and programs that are built to use those protocols, such as rsync. Rsync's advantage over normal FTP transfer is that it is able to transfer just the difference between the local and remote copy over the network. On the first request, this does not yield any advantage. On the second, however, while the FTP would transfer the whole file again, rsync only transfers the difference. There are several mirroring tools developed for mirroring. Some can handle mirroring through HTTP, when one does not need any account on an original computer. This eases the process, but is not efficient if there is a lot of content to mirror.

Databases can be also mirrored behind the user interface. It is very rare that people can access a database directly over the Internet. Normally, the database is hidded behind an interface. Many databases have replication possibilities and they normally use proprietary protocols to do so.

Today, CDNs are mostly done with proprietary solutions, but it is likely that open software solutions for overlay approach will appear.

3.3 What material should be decentralized

The one of the hardest questions to answer is "what content should be decentralized?" If everything is decentralized, it is difficult to keep everything updated. Decentralizing content that is not regulary used is expensive if decentralization is done for load balancing purposes. If it is done for usability purposes, it might be useful.

If the content is built in real time, there is no reason to decentralize it because it might be different on next request [39, p. 207]. This problem also depends on the decentralization method used. In the case of caching, only the content that is used gets decentralized, but the content provider cannot cannot au-

thoritatively control this kind kind of decentralization, only express requests on how long they wish this content should be cached, if at all.

In the case of mirroring, one can choose the content that is to be mirrored to another location. The content provider can always decide what content to replicate and control all duplicates. Another solution is to offer instructions to web site administrators about mirroring content. In this case, the content provider does not have control over all of the duplicates.

Deciding what material to replicate is often a difficult choice. In most cases, the content provider can rely on statistics of usage of an old version of the content or other similar types of content.

3.4 When is the best time to decentralize

The moment when decentralization is implemented depends entirely on the situation. Caching is done all of the time, but mirroring must be done according to some timetable.

Normally, administrators try to do all large data transfers during night hours, when there is normally less time-sensitive traffic in the network. This can be used as a base solution if the data transfer is inside one time zone. However, the Internet itself spreads all over the world.

When transferring large amounts of data between continents, it is advantageous to take some measurements of the current status of the network. The sample must be long enough that the results are useful.

Normally, all replication solutions have the potential for a manual implementation. Thus, if someone publishes something at certain time, it can at the same time be mirrored to other places.

Chapter 4

Case study: Web University and Funet

CERN produces many video lectures every year. The lectures are stored on a media server and anyone can use them for non-commercial purposes. Web University pilot is also offering the lectures through its web pages to Finnish users. Some of the lectures are already mirrored to Funet, but the mirroring is done mostly by hand.

In this case, the lectures can be easily replicated using FTP because the content does not change after creation. There is therefore no need for a more complex protocol that would transmit only the changed part of the content, like rsync.

In first section of this chapter, I explain Webcast server properties and introduce some log file analysis. The second section explains the structure of the connection and introduces some statistical information about CERN's interface to the GÉANT network. Measurements of the connection are shown in the third section. The current mirroring system is explained in the fourth section, and the new recommendation in the last.

4.1 Webcast server

A 600 Mhz Pentium III™processor powers the multimedia server in CERN and it is equipped with 512 MB of RAM, 100 Mbps Ethernet connections and 363 GB of hard disk space, at the moment. More hard disk space will be added if needed. Multimedia content is served by RealServer™7.x with a licence for 300 simultaneous clients.

To gather information on the possible bottlenecks in the server, I spoke with

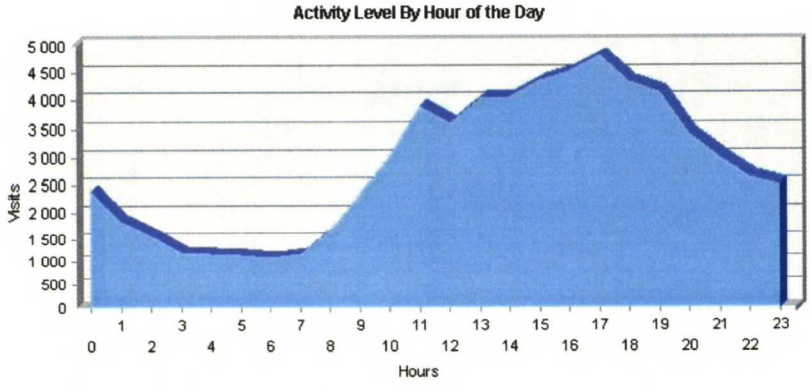

FIGURE 4.1: Visits during the day

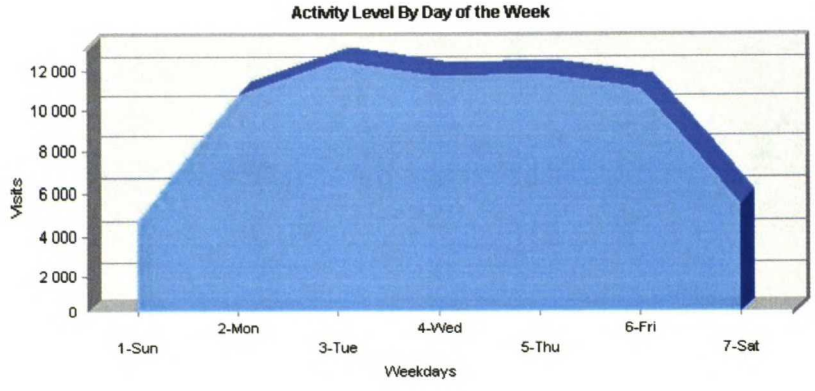

FIGURE 4.2: Visits during the week

CERN technicians and analysed the RealServer™log file with a trial version of Web Trends Analysis Series™7.0. The sample period of the log file started 3.9.1999 and ended 28.6.2002. Through conversations with the technicians, it became quite clear that there should be no problems with the server, but I wanted confirmation. The main thing that was pointed out was that the live broadcasts are what pushes the system to its limit, and then the bottleneck is the server's network connection. They can cope with this by lowering the video quality or upgrading the network interface.

Statistical analysis based on the log file showed that there are approximately 76 visits to the server each day, and one visit uses the resources slightly less than 13 minutes. 73 % of the visits occur between 10:00 and 23:00, as shown in figure 4.1 and 84 % of the visits occur between Monday and Friday, as shown in figure 4.2. 1.6 % of the visits come from Finland.

I used the Erlang model to calculate the probability that all 300 licences would be reserved at the same time (B_t). Because of the Erlang model's PASTA property, this is exactly the same probability as the probability that an arriving

request will be dropped because of a lack of licences (B_c).

Variable	Value	Definition
λ	0.05 users/min	Average arriving intensity of users
μ	12.8 min	Average time a user uses the service
n	300	Number of licences in CERN RealServer™
a	$\frac{\lambda}{\mu}$	Traffic intensity

TABLE 4.1: Variables in Erlang model

$$B_t := P\{X = n\} = \pi_n = \frac{\frac{a^n}{n!}}{\sum_{j=0}^{n} \frac{a^j}{j!}}$$

$$B_t := P\{X = 300\} = \pi_{300} \approx 0 \Leftrightarrow B_c \approx 0$$

According to these results, decentralization of the lectures to Funet does not give any remarkable advantage. Even if the usage of the lectures would triple and the length of average usage would double, there is no real advantage of decentralization. Another result is that if decentralization is to occur, the best times for transferring the desired content are between 3:00 and 7:00 Central European Time, and if there is a very large amount of content to be transferred, it should be transferred during the weekend.

4.2 Connection

The multimedia server at CERN is connected to the LAN through a 100 Mpbs connection. The connection from CERN to Funet goes through two research networks: GÉANT[1] and NORDUnet[2]. Illustrations of the networks are shown in Appendix A and Appendix B.

GÉANT is the name of the pan-European research and education network. Nine of the core circuits of GÉANT operate at 10 Gbps, while eleven others operate at 2.5 Gbps. It is currently the most advanced network of its kind worldwide in terms of speed. It connects more than 3,000 research and education institutions in 32 countries through 28 national and regional research and education networks. It has two 2.5 Gbps circuits in cooperation with North American research networks for connecting to other research networks around the world. It is financed by the European Commission. [46]

NORDUnet is the name of the Nordic national networks for research and education. It also connects these networks to the rest of the world, for example to

[1]The pan-European Gigabit Research Network.

[2]The Nordic Internet highway to research and education networks.

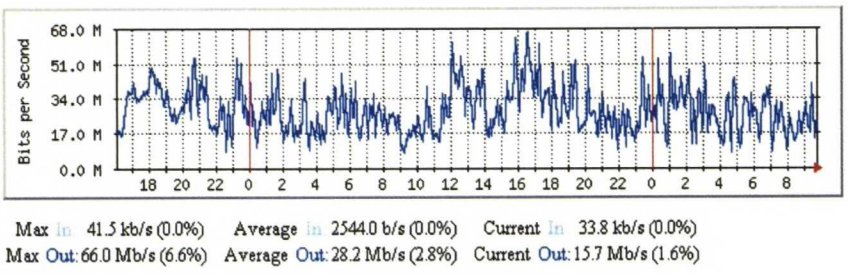

FIGURE 4.3: Monthly graph (5 minute average) of GÉANT connection

GÉANT. The core circuits of the NORDUnet operate at 2.5 Gbps. Network services offered by NORDUnet are based only on the Internet Protocol and these services are provided by a combination of leased lines and Internet services provided by other international operators. The Nordic countries finance it. [47]

I was not able to obtain much traffic-related information from these networks, so I must think of them more or less as a black box. The only information I could get these networks was the connection throughput between the networks, as shown in table 4.2.

Connection	**Throughput**
CERN - GÉANT	1 Gbps
GÉANT - NORDUnet	2.5 Gbps
NORDUnet - Funet	2*2.5 Gbps

TABLE 4.2: Connection from CERN to Funet

CERN provides statistical graphical information of the GÉANT connection. There are graphs of the connection, showing the daily (figure 4.3), weekly (figure 4.4), monthly (figure 4.5) and yearly [48] traffic. The graphs were taken on Wednesday, 26 June 2002 at 9:59. From these figures, one can see that there are no permanent problems with throughput for this connection. Throughput consumes, on average, less than 10% of the possible capacity of the connection.

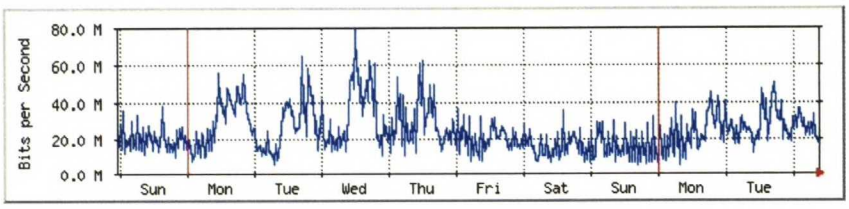

FIGURE 4.4: Monthly graph (30 minute average) of GÉANT connection

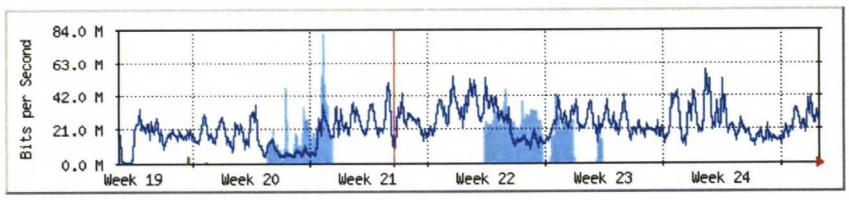

FIGURE 4.5: Monthly Graph (2 Hour Average) of GÉANT connection

4.3 Measuring the connection

4.3.1 Web lecture parameters

When using a recorded lecture from CERN multimedia server, one must retrieve the content from the server using a RealVideo™capable client software, such as RealPlayer™. This traffic differs from traffic produced from normal web browsing because most of the traffic is sent by UDP and is highly unidirectional.

To find the aspect ratio of UDP and other protocols, I recorded packet headers of four different sample lecture recordings. I used two different formats (Smil and RealVideo™) that are used in CERN. The first two samples were recorded completely. With the last two, I tried to create more control traffic, so I took two 20-minute samples where I wound forward every 5th minute.

For capturing the traffic that the samples produced, I used a WinDump-program on the same machine that ran the RealPlayer™. The operating system on that computer was Windows 2000™. WinDump is the porting of the tcpdump program that runs in a Unix environment [49, p. 139]. It prints out the headers of packets on a network interface so packets can be sorted with regular expressions [50]. The parameters I used with the WinDump program are listed in table 4.3.

Parameter	Explanation
-w	Write the raw packets to file from where they can be printed out with -r parameter
-r	Read raw packets from a file that was created with -w parameter
-i	Defines the used interface
-l	Buffer stdout line. Useful when I wanted to see the data while capturing it
-p	Interface in not promiscuous mode (Read only packets that are sent to this network interface)
-i	Defines the used interface

TABLE 4.3: Used tcpdump parameters

In order to analyse the traffic dump, I moved the dumps to a Linux environment, where I could use shell tools like grep and wc to search for more detailed information on the traffic. Results are shown in table 4.4.

Format	Throughput	Length	Packages	UDP packages	UDP/%
RealVideo™	180 Kbps	0:40:15	75549	75355	99,7
RealVideo™	40 Kbps	0:20:00	18992	18860	99,3
SMIL	180 Kbps	0:20:00	42578	42275	99,3
SMIL	100 Kbps	1:00:51	109096	108712	99,6

TABLE 4.4: Lecture information

In mirroring, the files are transferred using FTP because, as previously mentioned, the files are not going to be updated. To be able to simulate an FTP transfer, one must know the average size of a transferred lecture. I made an assumption that the RealVideo™file is almost equal to the size of the whole presentation. I located every agenda related file on the CERN media server that had an rm prefix and calculated the average size of the files. The result and number of samples are shown in table 4.5.

Samples	Average file size
176	166 MB

TABLE 4.5: Average RealVideo™file size

4.3.2 Measurements

For measurement, I was able to use the CERN Webcast server and a test machine in Funet. The Webcast server is described in section 4.1. The test machine in Funet is powered with a 200Mhz Pentium MMX™processor and is equipped with 64MB of RAM. The Ethernet connection is a full duplex 100 Mbps and there is 2,3 GB of hard disk.

For measuring the connection from the CERN Webcast server to Funet, I used an iperf program. To validate the iperf results, I ran transfer tests with iperf and Test TCP, as shown in table 4.6. The results are close enough that we can assume that iperf is a valid program for the tests.

Program	Average $\bar{x}$	Sample deviation $\bar{\sigma}$
iperf	18,47 s	0,24 s
TTCP	18,42 s	0,12 s

TABLE 4.6: Comparing iperf and TTCP results

Iperf is a command line program that produces statistical information of the measured connection. It can be run in client or server mode. The client sends required content to the server program, which produces the statistics. I used cron daemon in the CERN Webcast server to periodically run the iperf client. The server process runs in daemon mode on a Funet test machine.

I parsed the statistics with command line tools like grep and gawk. I transferred the results to Microsoft Excel™to make them more convenient.

I decided to measure the connection using UDP and TCP traffic. UDP traffic simulates normal lecture usage, and TCP simulates mirroring and other time independent deliveries.

I used the -W option for optimizing the window size on TCP test. The normal user does not have the skills to change the window size manually, and it would also create too many testcases to search for an optimal window size. Iperf could not provide the calculated values of optimal window size, so there is no window size parameter on the TCP measurement. All UDP tests were made using 64 KB buffers and 1470 byte datagrams.

- UDP test 1
  In the first measurement, I searched the boundaries for how many simultaneous connections and what throughput is possible through the connection. I used 5 different numbers of simultaneous users (1, 2, 4, 8, 16 users) and 5 different speeds (40 Kbps, 100 Kbps, 180 Kbps, 358 Kbps and 2 Mbps). Each run took 5 minutes. I decided not to use more simultaneous users because in practice, there has been an average of less

Parameter	Explanation
-s	Run the process in server mode
-D	Run the server as a daemon (only with -s parameter)
-c <host>	Run the process in client mode and connect to <host>
-u	Use UDP (TCP is the default)
-p	Port to listen on or connect to
-t	Time in seconds to transmit for (default 10 sec)(only with -c parameter)
-P	Number of parallel client threads (only with -c parameter)
-b	Requested throughput in bits/sec (default 1 Mbps)(only with parameters -c and -u)
-n	Number of bytes to transmit (only with -c parameter and not with parameter -t)
-W	Iperf will start with a default window size and performs a search for the optimal window size

TABLE 4.7: Used iperf parameters [51]

than one user from Finland per day. I chose throughput speeds according to the material produced at CERN: three lower throughputs and two higher ones to represent for for future usage.

I ran this test three times on different days. It took just over 2 hours to complete the test. According to the results, the network connection could easily fulfil the requirements for multimedia content transfer on all speeds that CERN uses today, even with 16 simultaneous users (see figure 4.6). Only the combination of 180 Kbps with 16 users had problems with lost datagrams, but it was still able to fulfil the throughput requirements. Even the 358 Kbps functioned flawlessly with 8 simultaneous users. With 16 users 358 Kbps dropped packages about 1 % of the time, see figure 4.8. The 2 Mbps caused problems with throughput and lost datagrams if there were more than 4 simultaneous users. According to the values from table 2.1, jitter was not a problem with any of the tests, see figure 4.7.

According to the results from this UDP test 1, the network connection can easily fulfil the requirements for CERN-produced video lectures if the quality of the lectures remains the same. With higher qualities, such as 358 Kbps or 2 Mpbs, problems arise if there are many simultaneous users in the system.

- UDP test 2
  In the second measurement, I investigated whether there are any major

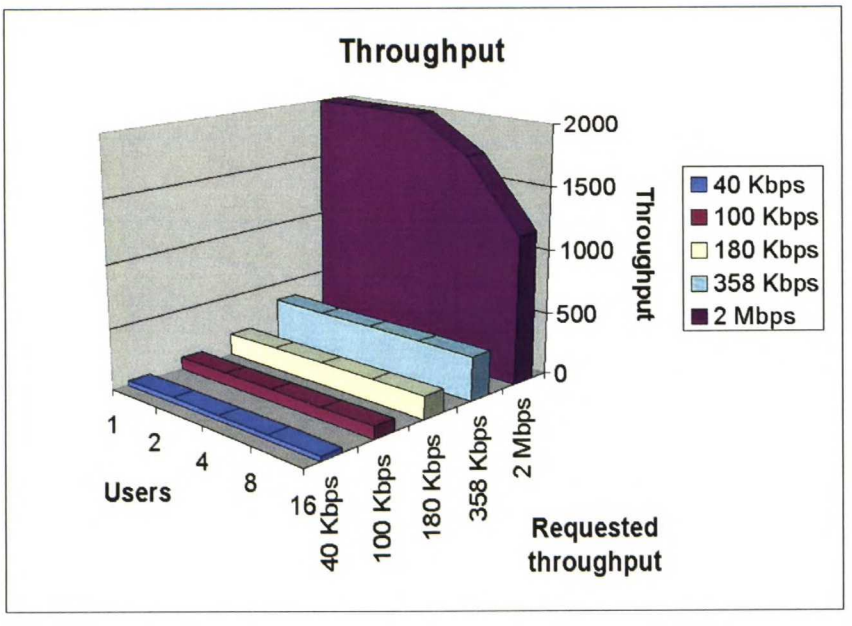

FIGURE 4.6: Throughput with different amounts of simultaneous users and different speeds

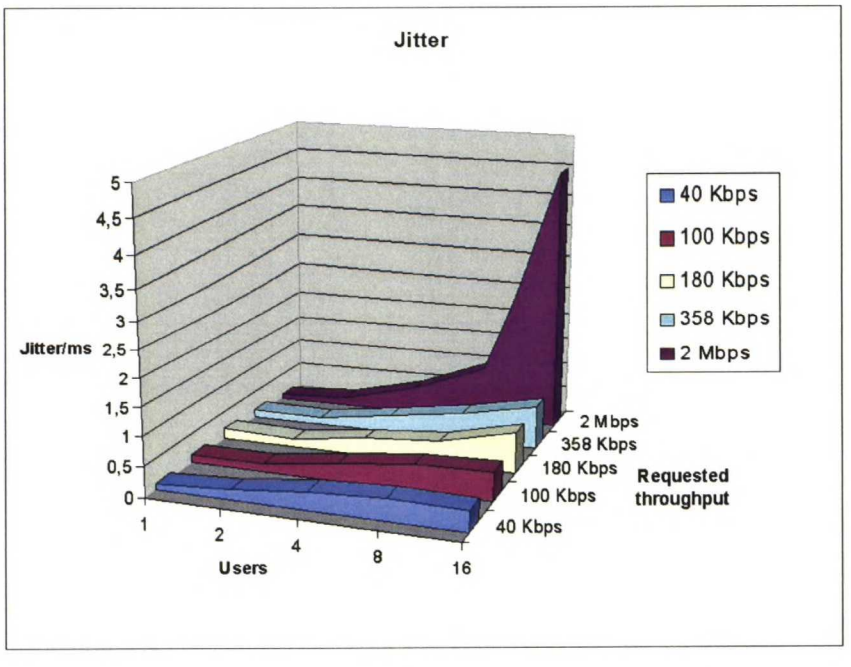

FIGURE 4.7: Jitter with different amounts of simultaneous users and different speeds

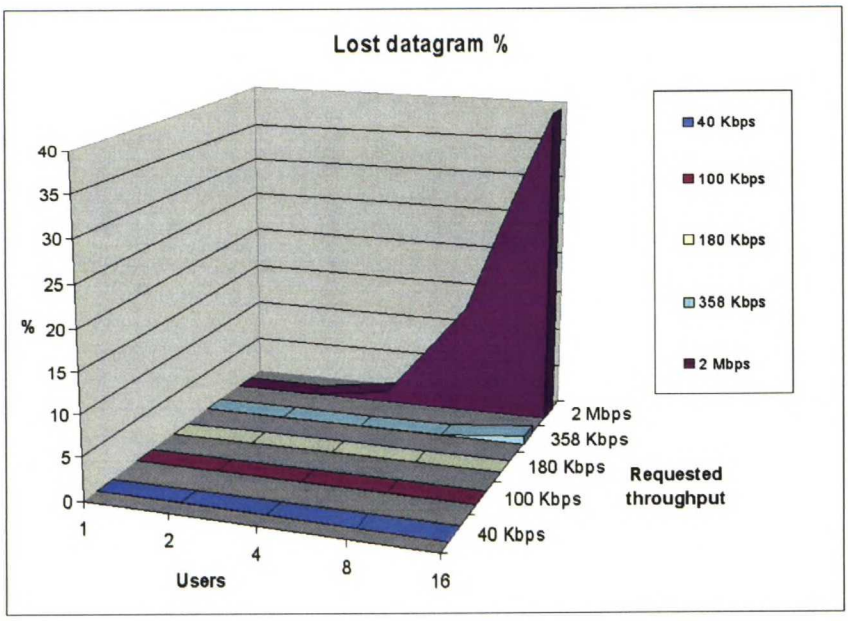

FIGURE 4.8: Lost datagrams with different amounts of simultaneous users and different speeds

differences in UDP throughput in a 24 hours period. The test period was one week, starting July 9. 2002 10:43 and ending July 16. 2002 11:13. The test performed 4 parallel transfers every two hours. Each connection lasted 30 minutes, and the required throughput was 180 Kbps.

The three main characteristics for this test were throughput, jitter and lost datagrams. The throughput stayed at the requested level through nearly the whole test period. Only once did it drop down to 179 Kbps from 180 Kbps. This was on Monday July 15. 2002 16:43-17:13. At the same time, there was also almost 0.5 % loss of datagrams sent, see figure 4.9. Aside from that period, datagram loss was 0.02 ‰. Jitter stayed under 0.5 ms through the whole test period, see figure 4.10.

Characteristic	**Average**
Throughput	179,99 Kbps
Jitter	0,35 ms
Lost datagrams	0,1 ‰

TABLE 4.8: Average characteristics from UDP test 2

According to the results from UDP test 2, the network connection can fulfil the requirements for today's video lectures at any time of the day. Only once during the week were there some problems with the quality of service, but the test produced much a higher usage of the network

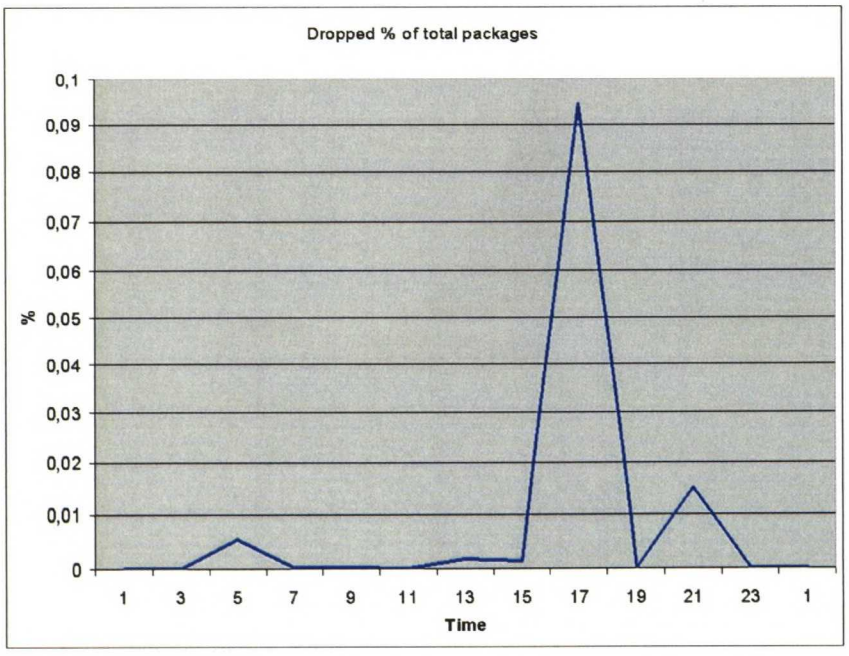

FIGURE 4.9: Datagrams lost during the 24 hours

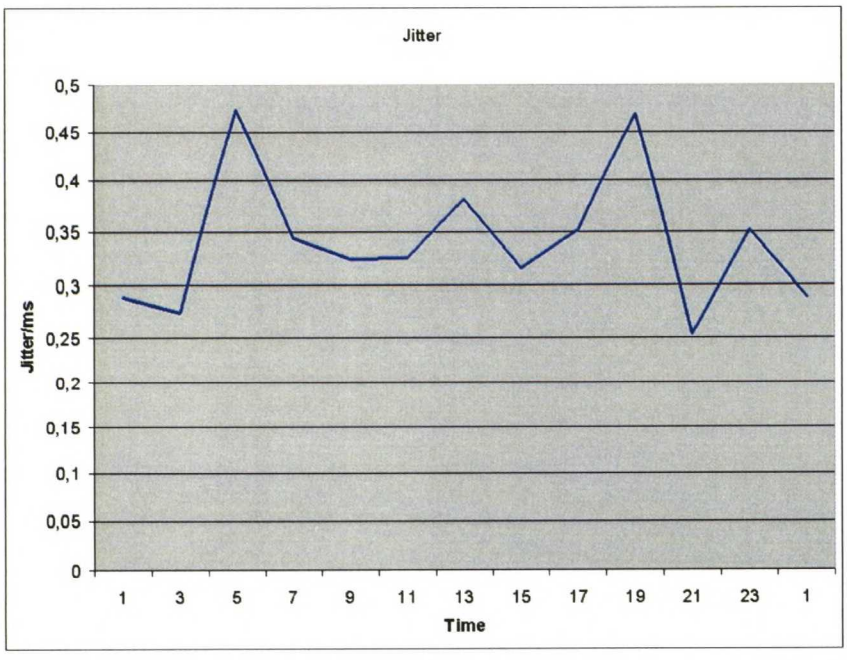

FIGURE 4.10: Jitter during the 24 hours

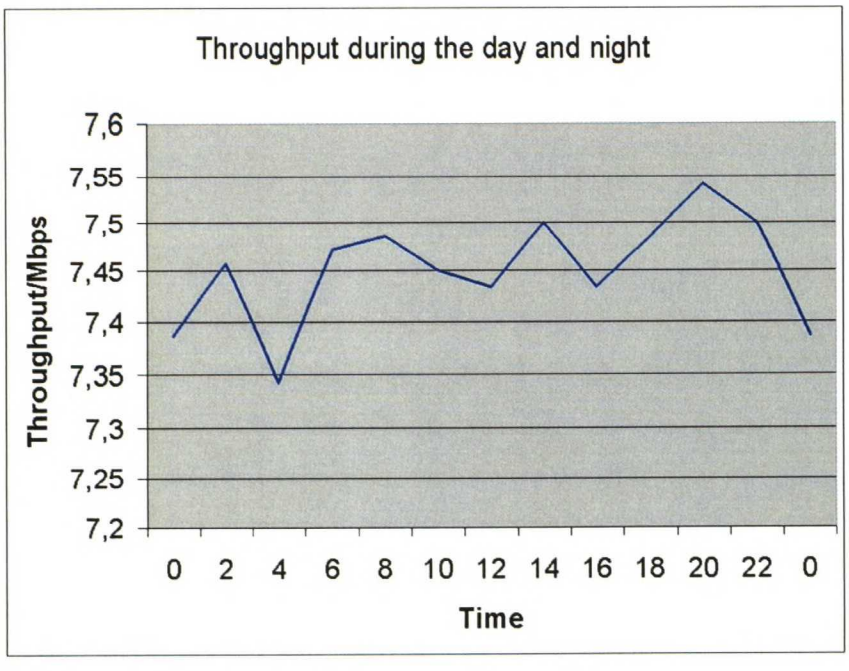

FIGURE 4.11: Throughput of transfer during the 24 hours

than Finnish users do, according to the logs from the Webcast server (see section 4.1).

- TCP test
  In the TCP measurement, I investigated whether or not there are any major fluctuations in TCP throughput during the 24 hours. The test period was one week, starting July 9. 2002 at 10:23 and ending July 16. 2002 at 10:27.

  The two main characteristics for this test were throughput and duration. They are actually the same thing, as one can see from the figures 4.11 and 4.12. For me, however, it was much easier to think about the duration of the transfer, rather than the throughput when comparing the results. Duration varied an average of less than 8 % during the whole test period and day averages only under 2 % from the test averages. The averages are shown in table 4.9. The longest transfer took 206.6 seconds to complete and the shortest took 187.8 seconds.

Characteristic	Average
Throughput	7.46 Mbps
Duration	191.5 sec
Size of the file	170 MB

TABLE 4.9: Average characteristics from TCP test

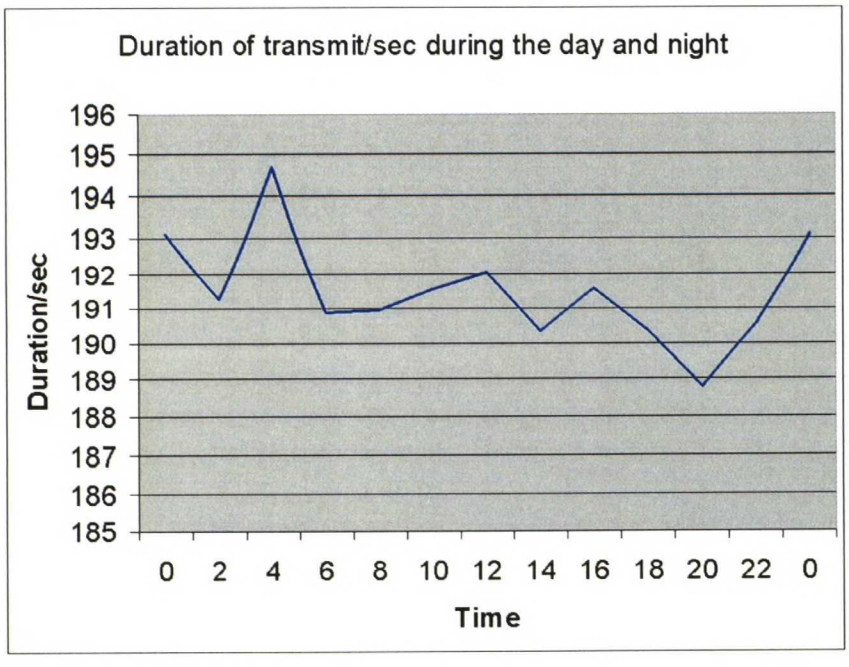

FIGURE 4.12: Duration of transfer during the 24 hours

According to the results from this TCP test, the timing of the mirroring is irrelevant if there are only one or two lectures to mirror. However, if one combines this with the results from figure 4.2 and figure 4.3, it seems that the best time to mirror lectures is between 4:30 and 6:30.

4.4 Existing solution

Some work has already been done on this case. There are a couple of scripts that manipulate some log files and copy certain directories. It is a semiautomatic system, where the Webcast server has a dedicated user for mirroring that has a special directory within its home directory.

4.4.1 Behaviour

In the mirroring process, one must first copy the lectures that one wants to mirror to the special directory in the Webcast server. The next night, Funet's media server copies all the content from that directory and updates the mirror log file in the Webcast server. All transactions are made using FTP.

One must then update the list of mirrored lectures, and this is done with a script executed from the command line. After that, a user of the web page is

able to see if the lecture is mirrored in Finland.

4.4.2 Problems and bottlenecks

The current system is not very easy to use and is not fully automated. There have to be user actions in every phase of the mirroring process. The user must choose what he/she wants to mirror and then manually update the mirroring links to the mirrored file. The biggest problem, however, is that one must decide in advance which lectures the system should mirror.

4.5 New solution

According to the measurements, decentralization is unnecessary for users in Funet. Most of the Finnish users of these lectures are able to use Funet's high-speed connections that are available to researchers. There also may be people that are interested in these lectures, but cannot use Funet's international connections, such as Finnish high school teachers and students. This group normally uses services provided by local ISPs. The connections inside Finland are normally capable of providing enough resources for allowing access to the lectures. Problems arise when using international connections, which can be congested. CERN routes all traffic to or from commercial networks through a gateway in the USA. The gateway in the USA can sometimes be a bottleneck. For this reason, I will recommend a simple system that can be used to decentralize part of the lectures to Funet.

4.5.1 Planning

The new solution needs to automatically make the decision of what to replications. It must also be easy to use and portable, so if somebody else wants to mirror lectures from CERN, they can quickly integrate the system to their own system. "Easy to use" implies clearly-defined interfaces and good error handling. Portability can be achieved by using a portable scripting environment or portable programming language and by dividing the task into smaller blocks. Each of these blocks should have its own simple task and be easily replaced with a better version without affecting other parts of the program.

Only two different recording types, RealVideo™and SMIL will be replicated. SMIL recording actually includes the RealVideo™file. Sync-O-Matic files are not replicated because they have a lot of absolute paths that must be converted before use. There are also some problems with the video on Sync-O-Matic that would need to be solved beforehand.

The system must be usable through HTTP and command line. The syntax of the external interfaces is shown in table 4.10. This solution leaves the real user interface open to the system. When the CERN metadata interface is ready and integrated with Funet's database, it will be easy to build the interface for this system.

Input	**Output**	**Type**
http://localserver/bin/redirect.pl? pathToLecture	newPathToLecture	HTTP
redirect.pl pathToLecture	newPathToLecture	Command line
Input example http://localserver/bin/redirect.pl?http://webcast.cern.ch/path/a99000s1t1.smi		
Output example http://webcast.cern.ch/path/a99000s1t1.smi		

TABLE 4.10:

The base of the system is built on two main modules: the redirector and replicator. These have three supporting modules: the appender, database interface and localPath, see figure 4.13. All modules except the localPath are located in the Funet server. localPath is located in Webcast server.

All interactions with the database are done through a database interface that is implemented in one file. The database system stores the information of mirrored lectures. The database uses a simple key pair consisting of the original URL and local URL.

The database is stored in a text file for portability reasons. File handling is easy to implement, and all functions are included in a file that can be later modified if the database implementation changes. With just a few simultaneous users and mirrored lectures, there is no need to build a new database for it. The interface to the database is simple and well-described, so if there are later problems with performance, it should be easy to replace the current text based system with a different one.

4.5.2 Behaviour

The main idea is to make a simple redirector, which redirects the client's request to local copy, if one exists. A first-time request is redirected to original address and the address is stored in the database. The next time, the replication process is started; it will retrieve the content from Webcast server and store the local address to the database. The next time someone requests that same address, they will be redirected to local copy.

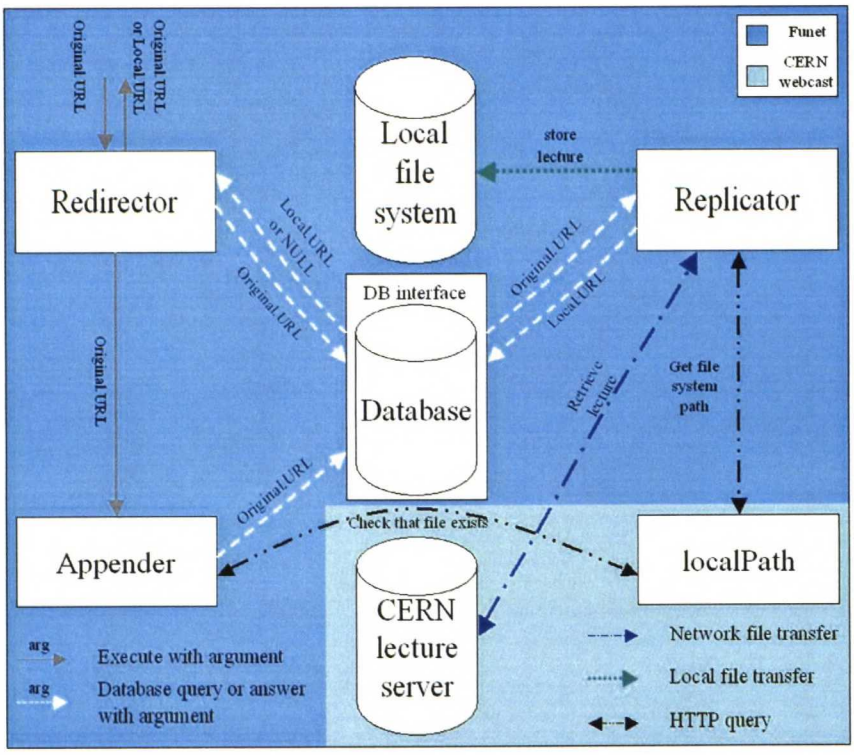

FIGURE 4.13: Block diagram of the mirroring solution

The system is split into five modules that each have their own tasks. The interfaces between the modules are meant to be simple and easy to change, if needed. Interactions are shown in figure 4.13 with arrows, and they are described in detail later in this subsection.

All other modules in the Funet server are using the database interface, so it will be best to explain that interface first. The code of the interface can be found in appendix G. It implements four functions that can be called from other local modules. Functions are explained here in alphabetical order.

- appenURL
  This takes the original URL of a lecture as a parameter and stores it in the database with NULL value as a key pair for this original URL. AppendURL does not return anything. It also takes care of the consistency of the database by locking the database file before using it.
- nullURLs
  NullURLs does not take any parameters. It returns an array that has original URLs of all the entries from the database that has NULL value as a local value. Because it only reads the database it does not lock the database file when using it.
- queryURL
  This takes the original URL of a lecture as a parameter and tries to find

a key pair for the URL. If a pair is found and is not NULL, that value is returned. If the pair is NULL, the original URL is returned. "NOT FOUND" string is returned if the pair is not found in the database. Because queryURL only reads the database, it does not lock the database file when using it.

- replaceURL
  ReplaceURL takes two parameters, the first being the original URL and the second being the local URL. It replaces the key pair of the original URL with the local URL. ReplaceURL does not return anything. It also takes care of the consistency of the database by locking the database file before using it.

The redirector module is the one that interacts with the lecture user. The user calls redirector with the URL to the CERN Webcast server as a parameter. It uses queryURL function from the database interface to check if there is a local replica of the lecture. The value function returned is stored in a variable. The function returns eitherthe URL or the "NOT FOUND" string. The URL can be the same as that which was queried or it can be local if it has been replicated. If the value of the variable is the "NOT FOUND" string, the redirector executes append module with the URL of the Webcast server as a parameter and stores the parameter URL to the variable. Then the user is redirected to that URL, or if the module is called from the command line, the URL is returned. The code for the redirector can be found in appendix C.

The appender checks that the parameter URL is syntactically correct. If the syntax is correct, it calls the localPath module from CERN to verify that the file actually exists. If the file is found in the Webcast server, the appender calls appenURL function with the URL as a parameter to store the URL in the database. The code for the appender can be found in appendix D.

The localPath module is used for fetching the file system path of a lecture. It takes two parameters: agenda number of the lecture and type of the recording. The syntax of an agenda number is well-defined. The format is always aXXXXXsXtX, where X is number from 0 to 9. It accepts only two types: ram and smi. It uses the "locate" system command to search the lecture from the server. If the file is found, it returns the path and name of the file. It actually gives back the first line of output of the locate command. If it is not found or the syntax of a query is wrong, it returns the "Not an agenda number" string. The code for the localPath can be found in appendix F.

The replicator is executed by some scheduling application. When its execution begins, it uses the nullURLs function to store all original URLs to an array. Each URL is processed separately. First, the agenda number and type of recording is parsed from the URL. The localPath module is called through HTTP using parsed values to get the file system path for the lecture. It makes a new directory with the name of the agenda number if that directory does not exist. Then it opens an FTP connection to the Webcast server and retrieves

the files that are needed for the type of lecture. It also checks to see if the video file has already been retrieved, because it is the only file that is included in both types of recordings that this system replicates. After retrieval, it closes the FTP connection and calls the database function replaceURL with the original URL and the new local URL as parameters. Then it starts the same procedure with next original URL in the array. It goes through all original URLs in the array. The code for the replicator can be found in appendix E.

4.5.3 Building

Four modules are made with perl[3], which can be obtained at no charge for many different operating systems [52]. It has powerful text-manipulation functions, is popular for programming World Wide Web electronic forms, and can be used as glue between systems, databases, and users. The localPath module is made with php[4] scripting language, which is available on the Webcast server.

All five modules have their own files: redirector.pl, append.pl, replicator.pl, database.pl and localPath.php, and their code can be found in appendices C, D, E, F and G. The database is stored in mirror.txt and also uses mirror.lock for preventing simultaneous writing to the mirror.txt file.

4.5.4 The future

This system will be available for anyone to use and develop further. If someone wants to use it, they need only get the code and an FTP account from the CERN Webcast server. At the moment, it only supports retrieval of the content. Deleting the content is now possible only by editing the mirrored.txt file and deleting the files from the file system. It would be a good idea to implement an interface for this or an automated system that deletes the content based on some rules. The system could use statistical information, which is easy to implement with a quicker database engine.

Web University's lecture archives are going to be upgraded to support this system later this year.

Cooperation with a metadata search engine will be implemented so that the metadata is fetched from CERN and stored in a local database in Funet. Finnish users can use Funet's database, which also has lectures other than CERN's, to research useful information. If a user wants to use SMIL or RealVideo™content from CERN Webcast, the query will go through the redirector. TUT is also interested in CERN's metadata, and they can also use the redirector in Funet.

[3]Practical Extraction and Report Language.

[4]PHP: Hypertext Preprocessor.

The system was developed to be easily upgradeable and, for the most part, does so. It is not optimised, so there is much to do if someone finds it interesting.

Chapter 5

Conclusions

Different multimedia types have different requirements for network connections. For replication, this is an even bigger problem. In this case, there was a need for good throughput and little jitter. The replicas could easily be done with simple FTP protocol, because there is no need for updating the replicas later.

Most of the work was done via studies of the Webcast server and the network connection. After the log analysis and some discussion, it became quite clear that the Webcast server is able to fulfil all current requests. No one can predict the future, but there are still available resources left on the server.

The connection measurements showed that the new GÉANT network, together with NORDUnet network, have amazing performance. An average-size lecture, which is 170 MB of data, can be transferred over TCP from CERN to Funet in an average of 3 minutes. The connection is even faster with UDP traffic. In the tests, there was no problem with 16 simultaneous connections with the highest quality that CERN provides today, although some problems arose with higher qualities. According to the log analysis, there is an average of only one user from Finland per day. This leaves plenty of capacity for the future.

Although there were no problems from CERN to Funet there are several potential users of CERN's material that do not have access to Funet's international connections. This includes, for example, researchers and students who are accessing the material from their homes. They are normally dependent on their local ISP's international connections. Local ISP's international connections are still congested, and CERN routes all traffic to or from commercial networks through a gateway in the USA.

To resolve this problem, I decided to develop a simple replication system. It is not a cache, nor is it a mirror. It is something in between. The main idea is to automate the replication process and make the system easy and portable. It will take the address of a CERN lecture and check if it has a local copy of

it. If no local copy exists, it will redirect the client to CERN and retrieve the lecture the next time the replicate process is executed. Thus, it only replicates the lectures that are queried from the replication system. The first query is always a miss, but the following queries will succeed if the replicate process has been executed between queries.

This recommendation is open for further development, and the example code is included in appendices C, D, E, F and G.

Bibliography

[1] Riitta Rinta-Filppula. Web university. `http://webuniversity.web.cern.ch/webuniversity/`, May 4 2002.

[2] Jack Grimes and Mike Potel. What is multimedia? *IEE Computer Graphics & Applications*, 11:49–52, January 1991.

[3] Dorée Duncan Seligmann. Just what is multimedia, anyway? *IEEE Multimedia*, 6:11–13, January–March 1999.

[4] Chwan-Hwa Wu and J. David Irwin. *Emerging Multimedia Computer Communication Technologies.* Prentice Hall, 1998.

[5] Juha Nurmela. Eurobarometer 50.1 inra and environmental attitudes2000. `http://www.tilastokeskus.fi/tk/yr/tietoyhteiskunta/`, May 16 2002.

[6] Judi N. Fernandez. *GIFs JPEGs & BMPs Handling Internet Graphics.* MIS:Press, Inc., 1997.

[7] James D. Murray and William vanRyper. *Ensyclopedia of Graphic File Formats.* O'reilly & associates, inc., 2 edition, 1996.

[8] Francois Fluckiger. *Understanding Networked Multimedia.* Prentice Hall, 1995.

[9] John Watkinson. *The Art of Digital Audio.* Focal Press, 2 edition, 1994.

[10] Fraunhofer-Gesellschaft. Basic about mpeg perceptual audio coding. `http://www.iis.fhg.de/amm/techinf/basics.html`, May 18 2002.

[11] Fraunhofer-Gesellschaft. Mpeg audio layer-3. `http://www.iis.fhg.de/amm/techinf/layer3/index.html`, May 18 2002.

[12] Dublin Core Metadata Initiative. Dcmi frequently asked questions. `http://dublincore.org/resources/faq/`, May 19 2002.

[13] Tom Badgett and Corey Sandler. *Creating Multimedia on Your PC.* John Wiley & Sons, INC., 1994.

[14] W3C. Synchronized multimedia integration language (smil) 1.0 specification. `http://www.w3.org/TR/REC-smil/`, May 18 2002.

[15] Charles Severance. Sync-o-matic. `http://www.netfact.com/syncomat/`, August 27 2002.

[16] Fred Halsall. *Data Communications, Computer Networks and Open Systems.* Addison-Wesley Publishing Company, 4 edition, 1996.

[17] Piyush Maheshwari. A dynamic load balancing algorithm for a heterogeneous computing environment. *System Sciences*, 1:338–346, 1996.

[18] Jukka Korpela. Standardi, mikä se on? `http://www.cs.tut.fi/~jkorpela/stand.html`, June 22 2002.

[19] Andrew S. Tanenbaum. *Computer Networks.* Prentice-Hall International Editions, 2 edition, 1989.

[20] Victor Reijs. Perceived quality of applications. `http://www.heanet.ie/Heanet/projects/nat_infrastruct/perceived.html`, August 15 2002.

[21] S. Deering. Rfc 1112 host extensions for ip multicasting, August 1989.

[22] J. Postel. Rfc 791 internet protocol, September 1981.

[23] J. Postel and J. Reynolds. Rfc 854 telnet protocol specification, May 1983.

[24] J. Postel and J. Reynolds. Rfc 959 file transfer protocol (ftp), October 1985.

[25] Andrew Tridgell and Paul Mackerras. *rsync man page*, 2.0 edition.

[26] J. Postel. Rfc 821 simple mail transfer protocol, August 1982.

[27] R. Fielding, J. Gettys, J. C. Mogul, H. Frystyk, P. Leach, and T. Berners-Lee. Rfc 2616 hypertext transfer protocol – http/1.1, June 1999.

[28] R. Khare and S. Lawrence. Rfc 2817 upgrading to tls within http/1.1, May 2000.

[29] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. Rfc 1889 rtp. a transport protocol for real-time applications, January 1996.

[30] H. Schulzrinne, A. Rao, and R. Lanphier. Rfc 2326 real time streaming protocol (rtsp), April 1998.

[31] Uyless Black. *Computer Networks Protocols, Standards and Interfaces.* Prentice Hall, 1987.

[32] J. Postel. Rfc 793 transmission control protocol, September 1981.

[33] J. Postel. Rfc 768 user datagram protocol, August 1980.

[34] Michael Santifaller. *TCP/IP and NFS Internetworking in a UNIX Environment.* Addison-Wesley Publishing Company, 1991.

[35] Liam B. Quinn and Richard G. Russel. *Fast Ethernet.* Wiley Computer Publishing, 1997.

[36] Brian D. Davison. A web caching primer. *IEEE Internet Computing*, 5(38–45), July–August 2001.

[37] Randal C. Burns, Robert M. Rees, and Darrell D. E. Long. Efficient data distribution in a web server farm. *IEEE Internet Computing*, pages 56–64, July/August 2001.

[38] Nikhil Chandhok. Web distribution systems : Caching and replication. `http://www.cis.ohio-state.edu/~jain/cis788-99/web_caching/index.html`, July 9 2002.

[39] Michael Rabinovich and Oliver Spatscheck. *Web Caching and Replication.* Addison-Wesley Publishing Company, 2002.

[40] Eljas Nikkilä. Vertailussa web-palvelinohjelmistot. *Tietokone*, pages 76–81, May 2002.

[41] M. Chantel. Rfc 1919 classical versus transparent ip proxies, March 1996.

[42] Darren Sanders, Mohit Aron, Peter Druschel, and Willy Zwaenepoel. Scalable content-aware request distribution in cluster-based network servers. `http://citeseer.nj.nec.com/rd/31958753,290556,1,0.25,Download/http:qSqq%Sqciteseer.nj.nec.comqSqcacheqSqpapersqSqcsqSq13872qSqhttp:zSzzSzwww.cs.rice.e%duzSz~druschelzSzusenix00.pdf/aron00scalable.pdf`, August 19 2002.

[43] Daniel A. Menascé and Virgilio A. F. Almeida. *Capasity Planning for Web Services.* Prentice Hall, 2002.

[44] Irwin Lazar and William Terrill. Exploring content delivery networking. *IT Professional*, 3:47–49, July–August 2001.

[45] Primitivo Cervantes. Real-time remote data mirroring. *Sys Admin*, 10, June 2001.

[46] External Relations Manager. The pan-european gigabit research network. `http://www.dante.net/geant/geant-brochure-dec01.html`, June 26 2002.

[47] NORDUnet Information Service. What is nordunet? `http://www.nordu.net/services.html`, June 26 2002.

[48] Multi Router Traffic Grapher. Cern accessto geant. `http://sunstats.cern.ch/mrtg/geant.html`, June 26 2002.

[49] Loris Degioanni. Development of an architecture for packet capture and network traffic analysis. Graduation thesis, Politecnico Di Torino, March 2000.

[50] Van Jacobson, Craig Leres, and Steven McCanne. *tcpdump man page.* Lawrence Berkeley National Laboratory, University of California.

[51] Ajay Tirumala, Feng Qin, Jon Dugan, and Jim Ferguson. Iperf 1.6 - the tcp/udp bandwidth measurement tool. `http://dast.nlanr.net/Projects/Iperf/`, July 16 2002.

[52] Jarkko Hietaniemi. Cpan/ports. `http://www.cpan.org/ports/index.html`, July 18 2002.

Appendix A

The GÉANT Network

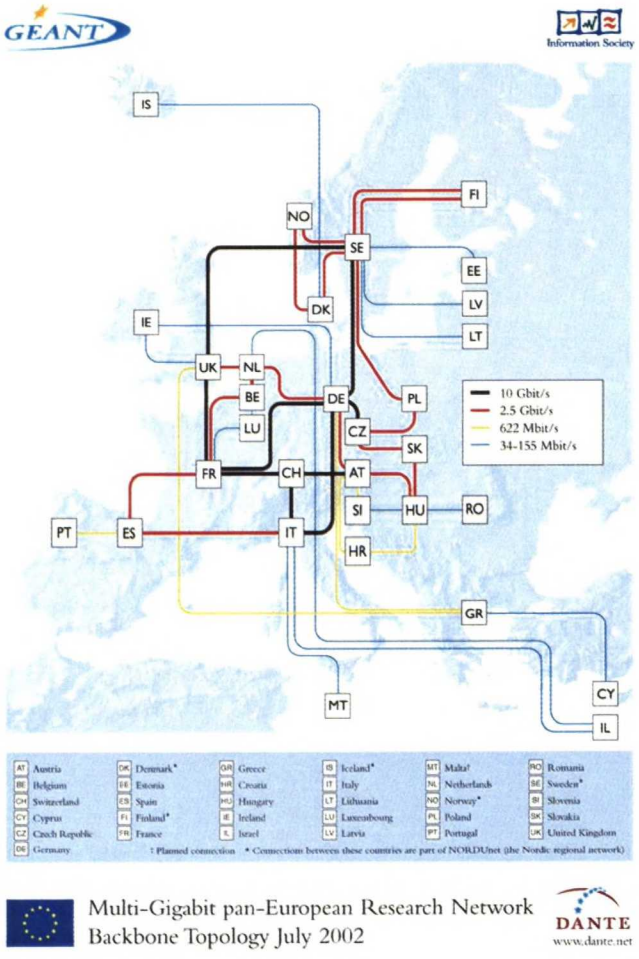

FIGURE A.1: The GÉANT Network used with permission from Dante

Appendix B

The NORDUnet Network

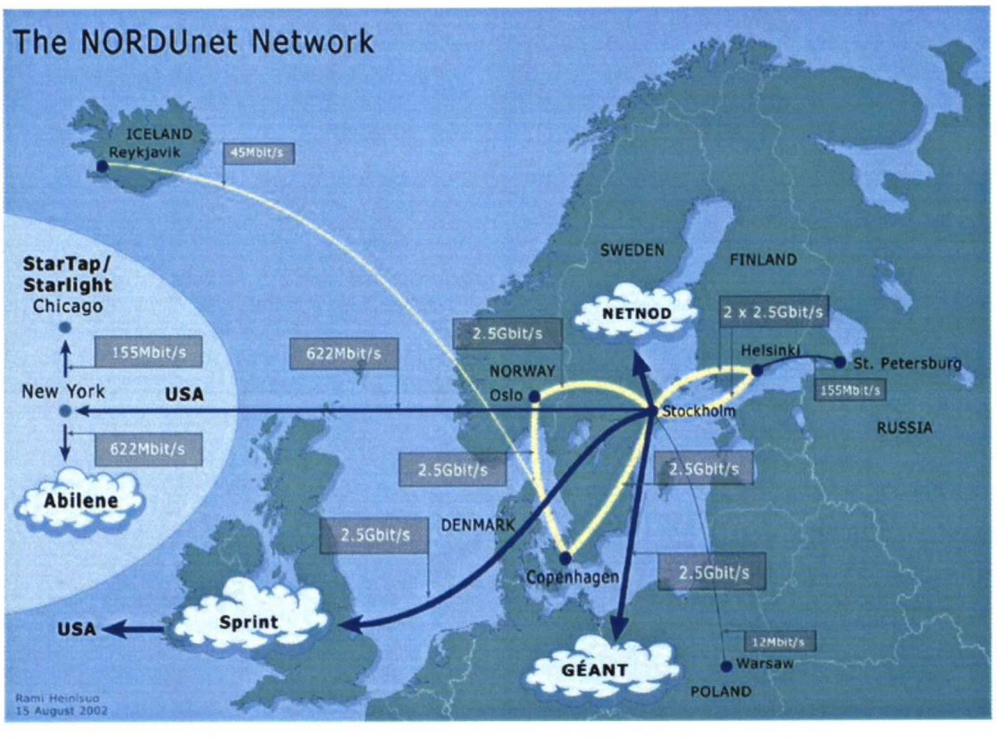

FIGURE B.1: The NORDUnet Network used with permission from NORDUnet

Appendix C

Redirector

redirect.pl

```
#!/usr/bin/perl
#####################
# Tuomo Koskenvaara #
# 13.9.2002 Geneva  #
# Version 1.0       #
# Licence GPL       #
#####################

use lib "/var/www/cgi-bin/";

# Load database functions
require "database.pl";

# Get URL from HTTP Header or from command line
$inputAddress = $ENV{'QUERY_STRING'}; # Input from url
if ($inputAddress eq '') {
  $inputAddress = @ARGV[0];          # Input from command line
}

# check that URL starts with "http://webcast.cern.ch" and
# ends with .ram or .smi and contains only valid characters.
if ($inputAddress =~
    /^http:\/\/webcast\.cern\.ch\/[\w\/.]+(\.smi|\.ram)$/x ) {

    # Check that URL does not contain .. for moving in directory tree
    $test = $inputAddress =~ /\.\./;

    if ((!$test) && (length($inputaddress)<512)) {
      # Fetch address from database
      $redirectedAddress = queryURL($inputAddress);
```

```
      if ($redirectedAddress eq 'NOT FOUND') {
        # Queried address is not in database
        # append.pl will store it there
        defined(my $childpid = fork) or die "Cannot fork: $!";
        if ($childpid) {              # parent does:
          $redirectedAddress = $inputAddress;
          # Redirect to address
          print "Location: $redirectedAddress\n\n";
        } else {                      # child does:
          exec ("pathToAppend.pl/append.pl", "$inputAddress");
      }
    }
  }
}
```

Appendix D

Appender

append.pl

```
#!/usr/bin/perl
#####################
# Tuomo Koskenvaara #
# 13.9.2002 Geneva  #
# Version 1.0       #
# Licence GPL       #
#####################

use LWP::Simple;

# Load database functions
require "database.pl";

# Get new url as an argument from redirect.pl
$newURL = @ARGV[0];

# check that URL starts with "http://webcast.cern.ch" and
# ends with .ram or .smi
if ($newURL =~
    /^http:\/\/webcast\.cern\.ch\/[\w\/.]+(\.smi|\.ram)$/x ) {

  ($agenda, $type) = $newURL =~ m/.*\/(\w+)\.(\w+)$/x;

  my $url = "http://webcast.cern.ch/AskPathFromWebcasPersonel/";
  $url = $url . "localPath.php?agenda=" . $agenda;
  $url = $url . "&type=" . $type;
  my $cernAddress = get $url;

  if ($cernAddress ne 'Not an agenda number') {
    appendURL($newURL);
```

```
  }
}
```

Appendix E

Replicator

replicate.pl

```
#!/usr/bin/perl
#####################
# Tuomo Koskenvaara #
# 13.9.2002 Geneva  #
# Version 1.0       #
# Licence GPL       #
#####################

use LWP::Simple;
use Net::FTP;

# local directory path
$localPath = "/var/www/html/";
$binPath   = "/var/www/cgi-bin/";

# Load database functions
require "database.pl";

my @id = nullURLs();

for ($i = 0; $i <= $#id; $i++) {
  my $originalURL = $id[$i];

# grep $agenda and $type from $original
  ($agenda, $type) = $original =~ m/.*\/(\w+)\.(\w+)$/x;

#############################################################
# Retrieve the real address of the file on servers file system
  my $url = "http://webcast.cern.ch/AskPathFromWebcasPersonel/";
  $url = $url . "localPath.php?agenda=" . $agenda;
```

```
  $url = $url . "&type=" . $type;
  my $cernAddress = get $url;

  ($path) = $cernAddress =~ m/(.*)\/\w+\.\w+$/x;


################################################################
# retrieve content by FTP

  # Local test directory to store the retrieved content
  chdir("$localPath");

  # Check if there is allready a directory for that lecture
  if (!( -e "$agenda")) {
    mkdir "$agenda";
  }

  # Go to agenda directory
  chdir("$agenda");

  # Open FTP-connection
  my $ftp = Net::FTP->new("webcast.cern.ch", Debug => 0);
  $ftp->login("username", "passwd");
  $ftp->cwd($path);

  # Files neaded for ram extension
  # 1. ram file (must be generated with right rtsp URL)
  # 2. rm file (the real video file)
  # Store files to to the folder named by agenda number.
  # If the rm file is allready been mirrored don't
  # mirror it again
  if ($type eq 'ram') {
    if (!( -e "$agenda.rm")) {
      $ftp->get("$agenda.rm");
    }
    # Make .ram file
    # path to the rtsp server
    my $rtspPath = "rtsp://localserver/$agenda/$agenda.rm";
    $file = "$agenda.ram";
    open(TXT, ">>$file");
    print TXT "$rtspPath\n";
    close(TXT);

    # Store the local address
    $localaddress = "http://localserver/$agenda/$agenda.ram";
  }
```

```
  # Files neaded for SMIL show
  # 1. smi file (skeleton of the SMIL presentation)
  # 3. rm file (the real video file)
  # 3. text.rt
  # 4. subtitle.rt
  # 5. links.rt
  # 6. slides.rp
  # 7. slides directory with its contents
  # 8. http://cds.cern.ch/img/cds.gif
  # Store files to to the folder named by agenda number.
  # If the rm file is allready been mirrored don't
  # mirror it again
  if ($type eq 'smi') {
    $ftp->get("$agenda.smi");
    if (!( -e "$agenda.rm")) {
      $ftp->get("$agenda.rm");
    }
    $ftp->get("text.rt");
    $ftp->get("subtitle.rt");
    $ftp->get("links.rt");
    $ftp->get("slides.rp");
    mkdir "slides";
    chdir("slides");
    $ftp->cwd("slides");
    @dirList = $ftp->dir();
    foreach (@dirList) {
      ($file) = m/.*\s(\w+\.gif)$/x;
      if ($file ne '') {
        $ftp->get($file);
      }
    }
    $localaddress = "http://localserver/$agenda/$agenda.smi";
  }
  # Close FTP connection
  $ftp->quit;

#####################################################################
# Update mirrored.txt file
  chdir($binPath);
  replaceURL($originalURL, $localaddress);
}
```

Appendix F

Localpath

localPath.php

```
<?
//////////////////////
// Tuomo Koskenvaara //
// 13.9.2002 Geneva  //
// Version 1.0       //
// Licence GPL       //
//////////////////////

// Returns file system path if the agenda number is
// correct and the file exist otherwise
// return "Not an agenda number".

// $agenda - Agenda number of the lecture
//           Syntax of an agenda number aXXXXXsXtX
//           where X is a number from 0 to 9
// &type   - Type of a file (rm or smi)

if (preg_match ("/a\d{5}s\dt\d/", $agenda)) {
  $command = "locate $agenda.$type";
  $output = shell_exec ($command);
  if ($output == '') {
    print "Not an agenda number";
  } else {
    $address = preg_split ("/[\s,]+/", $output);
    print $address[0];
  }
} else {
  print "Not an agenda number";
}
?>
```

Appendix G

Database interface

database.pl

```
#!/usr/bin/perl
#####################
# Tuomo Koskenvaara #
# 13.9.2002 Geneva  #
# Version 1.0       #
# Licence GPL       #
#####################

sub appendURL {
  my $newURL = $_[0];
  my $file = "mirrored.txt";
  my $lockFile = "/tmp/mirror.lock";

  # Wait until lockFile has been removed
  # Replicator or other instance of
  # appender can put this lockfile on
  # when it manipulates with the file
  while( -e $lockFile) {
    sleep 1
  }

  # Lock database
  system "touch $lockFile";

  # Store new address to database
  open(TXT, ">>$file") || die "Can't open $file: $!";
  print TXT "$newURL\n";
  print TXT "NULL\n";
  close(TXT);
```

```
  # Remove lock from database
  system "rm -f $lockFile";
}

sub queryURL {
  my $queriedURL = $_[0];
  my $file = "mirrored.txt";

  # Fetch addreses from the database to URLs array
  open(TXT, $file) || die "Can't open $file: $!";
  my %URLs = <TXT>;
  close(TXT);

  # test variable, if address is already
  # stored in database, but it has not yet
  # been mirrored
  $foundURL = 0;

  foreach $original (keys %URLs){
    # Some stripping for comparing
    $origTmp = $original;
    $origTmp =~ s/\s+//g;

    # Comparing input address and addresses in database
    if ($origTmp eq $inputAddress) {
      $localAddressTmp = $URLs{$original};
      $localAddressTmp =~ s/\s+//g;
      if ($localAddressTmp ne 'NULL') {
        # Found match and the local address is
        # not NULL
        return $URLs{$original};
      } else {
        # Found match but the local address is NULL
        # so no redirection.
        return $original;
      }
    }
  }
  # Queried address is not in database
  return "NOT FOUND";
}

sub replaceURL {
  my $originalURL = $_[0];
  my $localURL    = $_[1];
  my $file        = "mirrored.txt";
  my $lockFile    = "mirror.lock";
```

```
  # Wait until lockFile has been removed
  # Replicator or other instance of
  # appender can put this lockfile on
  # when it manipulates with the file
  while( -e $lockFile) {
    sleep 1
  }

  # Lock database
  system "touch $lockFile";

  # Store mirrored.txt to URLs variable
  open(TXT, $file) || die "Can't open $file: $!";
  my %URLs = <TXT>;
  close(TXT);

  #replace NULL with local when original found;
  foreach $original (keys %URLs){
    # Comparing OriginalURL and addresses in database
    if ($original eq $originalURL) {
      $URLs{$original} = "$localURL\n";
    }
  }

  # Store modified URLs variable to mirrored.txt
  open(TXT, ">$file") || "Can't open $file: $!";
  print TXT %URLs;
  close(TXT);

  # Remove lock from database
  system "rm -f $lockFile";
}

sub nullURLs {
  my $file = "mirrored.txt";

  # Fetch addreses from the database to URLs array
  open(TXT, $file) || die "Can't open $file: $!";
  my %URLs = <TXT>;
  close(TXT);

  foreach $original (keys %URLs){
    # Some stripping for comparing
    $origTmp = $original;
    $origTmp =~ s/\s+//g;
```

```
    $localAddressTmp = $URLs{$original};
    $localAddressTmp =~ s/\s+//g;
    if ($localAddressTmp eq 'NULL') {
      # Found a local address that is NULL
      push (@id, $original);
    }
  }
  return @id;
}

1;
```